

[19]中华人民共和国专利局

[11]公开号 CN 1035373A



[12] 发明专利申请公开说明书

[21] 申请号 88107804.2

[51] Int.Cl⁴

G06F 13/14

[43] 公开日 1989年9月6日

[22] 申请日 88.11.14

[30] 优先权

[32] 87.11.18 [33] US [31] 122292

[71] 申请人 国际商用机器公司

地址 美国纽约

[72] 发明人 威廉·爱德·哈默

瓦尔特·亨利·史文尼

弗兰里卡·约瑟夫·赛茨那

[74] 专利代理机构 中国国际贸易促进委员会专利

代理部

代理人 刘建国

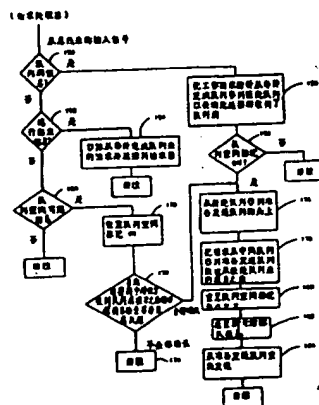
说明书页数: 42

附图页数: 29

[54] 发明名称 总线流程控制机构

[57] 摘要

描述在服务驱动的进程到进程通讯环境中的工作请求流程。进程与总线管理之间的逻辑连接把总线单元接到 I/O 总线上, 总线管理把这些逻辑连接分配给各连接群。每个总线单元有自己的连接群。把总线单元的资源分配给每个连接群, 并用一系列总线单元信息控制工作流程, 使没有更多资源的群不再接受进一步的工作请求。重新排列被拒绝的工作请求, 并在该连接群有可用资源时重新发送它们。



(BJ)第1456号

权 利 要 求 书

1. 控制涉及进程群之间信息的工作流程的一种方法，其中每个群属于一个松散耦合的分布式处理器网络中的一个处理器，且每个群中的各个进程分享处理器的资源，其特征为包括如下步骤：

由含有某一个群中的一个请求进程的处理器向另一个群中的一个服务进程发送一个信息，启动含有服务进程的处理器中的工作；

接收信息后启动服务处理器中的工作；

在服务处理器中识别资源不够的条件；

由服务处理器向请求处理器返回一个信息，指明资源不够；

当在该队列中资源可用时，服务处理器向请求处理器发送一个资源可用信息；

根据此资源可用信息，由请求处理器向服务处理器发送一个重新开始信息；以及

将信息排队，以便仅在从请求处理器收到重新开始信息后，在服务处理器中启动从请求处理器接收的工作。

2. 一个用于在松散耦合的分布式处理器网络中的总线单元的通讯管理机构，其中各总线单元是由一条总线耦合在一起的，此通讯管理机构方便了分别属于目标和源总线单元的各进程之间的进程到进程的通讯，其特征为在总线单元处的这种结构包括，

用于接收启动工作的信息的接收装置，所说的信息来自源总线单元；

一个与接收装置相匹配的资源管理，所说的资源管理包括，

管理在总线单元上预先指定的进程群的总线单元资源的管理装置；

显示“满”的装置，如果没有适当的资源可用于该群以完成该工作，则它根据启动预先指定的一个群中的工作的信息，向源总线单元返回一个队列满信息；

显示“资源可用”的装置，在资源对该群成为可用时，它向源总线单元发送一个资源可用信息；以及

工作接收装置，只要源总线单元首先提供一个重新开始信息，并且该群具有适当资源可用，它就从源总线单元接受一个信息来启动工作。

3. 如权利要求2所述的管理结构，其特征是该资源管理进一步包括，

根据启动工作请求的总线单元中的进程，向该第二总线单元发送一个启动工作的信息的装置。

4. 如权利要求3所述的管理机构，其特征是该资源管理进一步包括，

存储以前送出的启动第二总线单元中的工作信息的储备装置，所说的被存储的信息指明该第二总线单元和信息被送往的群；

向该第二总线单元发送一个重新开始信息的重新开始装置，该单元曾经用一个队列满信息响应来自该总线单元的一个启动工作信息，所说的重新开始信息是在从第二总线单元收到一个队列空间可用信息以后发送的；以及

与上述重新开始装置和储备装置相耦合的，重新发送启动工作信息的重新发送装置，该信息曾使第二总线单元发送一个队列满信息。

5. 如权利要求2所述的管理机构，其特征为，该显示“满”装置进一步包括监控对每个群仍然可用的资源的装置。

6. 如权利要求5所述的管理机构,其特征为,该监控装置包括对每个群的一个计数,该计数减少或增加取决于该群是否完成了一个工作请求或者该群是否收到了一个工作请求。

7. 如权利要求6所述的管理机构,其特征为,根据该计数值与该群最大值的比较,由上述显示“满”装置返回队列满信息。

8. 如权利要求7所述的管理机构,其特征为,每个工作请求使该监控装置把该计数增加或减少1。

9. 如权利要求7所述的管理机构,其特征为,每个工作请求包含一个该请求要求服务所需资源量的标志,并且其中监控装置依赖于此标志使该计数增加或减少一个量。

10. 如权利要求4所述的管理机构,其特征为该资源管理进一步包括,

一个为储存将由总线单元发送的工作请求的发送队列。

一个以原来发送的同样次序排列的储存待发送的工作请求的中间队列,对这些请求而言,它们要用的群已经收到了一个队列满信息;

如果工作请求要用的群已经收到一个队列满信息,则把该工作请求从发送队列传送给中间队列的装置;以及

检查储备装置的装置,它确定对于已经收到队列满信息的各个群,是否所有的工作请求都导致收到队列满信息。

11. 如权利要求10所述的管理机构,其特征为,一旦上述检查装置表明它们全都收到了队列满信息,该重新发送装置就重新发送那些收到过队列满信息的工作请求,所说的重新发送装置以它们原来发送的同样次序来发送它们,并且,然后所说的重新发送装置以它们原来在发送队列中排列的同样次序来发送中间队列上的工作请求。

12. 多处理器计算机系统的一种工作请求流程控制机构，其上多个进程在一个服务驱动环境下相互进行通讯，它包括，

一个第一总线单元，它具有一个主存储和执行第一进程的装置，

一个第二总线单元，它具有一个主存储和执行第二进程的装置，

以及

一个与所说的第一总线单元和第二总线单元相耦合的 I/O 总线。此 I/O 总线方便了该第一和第二总线单元之间的通讯；

其特征为，

所说的第一总线单元有一个总线管理，用来在上述第一和第二进程之间建立逻辑连接，并且将该连接与可用有限数量的第一总线单元资源的一个连接群联系起来；

所说的第二进程引发一个将由上述第一进程服务的工作请求；

所说的第二总线单元有一个总线管理，用来向第一总线单元的总线管理发送一个“运行开始”信息，通知第二总线单元由第二进程引发的该工作请求，所说的总线管理接收该“运行开始”信息，并确定用来服务于该工作请求的连接群；以及

在所说的第二总线单元的总线管理中确定该连接群是否有足够的资源可用来服务于该工作请求的装置。

13. 如权利要求12所述的工作请求流程控制机构，其特征为，所说的第一总线单元包括执行多重进程的装置，并且其中第一总线单元的总线管理根据第一总线单元中被连接进程的特征，在至少三个连接群之间安排各个连接。

14. 如权利要求13所述的工作请求流程控制机构，其特征为第一总线单元的总线管理进一步包括，

每个连接群的计数，用于保持跟踪在各自连接群中具有连接的进程所服务的各个工作请求；以及

与该计数相耦合的装置，用于当一个在连接群中具有连接的进程收到一个启动工作信息，而该连接群的计数具有超过预定数的值时，向第二总线单元发送一个队列满信息。

15. 如权利要求14所述的工作请求流程控制机构，其特征为第一总线单元的总线管理进一步包括：

与上述计数相耦合的装置，用于当该计数低于第二个预定值时，向第二总线单元发送一个队列空间可用信息，其中所说的发送队列满信息的装置，根据在已发送了第一个队列满信息的连接群中具有连接的一个进程收到启动工作信息，而连续发送队列满信息，直到根据队列空间可用信息从第二总线单元收到一个重新开始排队信息为止。

16. 如权利要求15所述的工作请求流程控制机构，其特征为第二总线单元的总线管理进一步包括：

接受队列满信息的装置；

确定启动工作信息原来次序的装置，这些信息使该连接群产生队列满信息，并且是在产生第一个队列满信息的“运行开始”信息之后发出的；

在确定了原来的次序并接收到队列空间可用信息后，对所说的连接发送一个重新开始排队信息的装置；以及

以原来的次序向第一总线单元发送所说的启动工作信息的装置。

17. 在由一根总线联系起来的具有多个总线单元的多进程服务驱动工作处理系统中控制工作请求流程的方法，其特征在于包括下述步骤：

a. 在处于不同总线单元上的请求和服务进程之间建立多个逻辑连接;

b. 将每个总线单元中的每个连接安排到多个逻辑连接群中的一个群中;

c. 使用与启动工作请求相连的总线, 通过逻辑连接向服务进程发送启动工作信息;

d. 在总线单元处接收执行一个进程的启动工作信息, 该进程在该启动工作信息中被确定为该工作请求的服务者;

e. 确定对该启动工作信息中被确定的服务进程有逻辑连接的连接群;

f. 跟踪每个连接群服务的工作请求的数目;

g. 确定所收到的启动工作信息对于确定的连接群是否超过预定的工作请求数。

18. 如权利要求17所述的方法, 其特征在于包括下述步骤:

如果对确定的连接群, 工作请求的数目超过了该预定数, 则向含有该请求进程的总线单元发送一个队列满信息。

19. 一种用于松散耦合的分布式处理器网络中的总线单元通讯管理机构, 其中各总线单元由一根总线联系起来, 该通讯管理机构方便了属于目标和源总线单元的各个进程之间的进程到进程的通讯, 其特征为在总线单元处的该机构包括:

接收启动工作的信息的接收装置, 所说的信息来自源总线单元;

一个与该接收装置相耦合的资源管理, 所说的资源管理包括:

对该总线单元上的预定的进程群管理总线单元资源的管理装置;

启动装置的信息。该装置用来接收从缺乏手段的源总线单元来的

工作，并且当一个群由于接收该信息而使资源超出界限时则拒绝启动工作的信息。

说 明 书

总线流程控制机构

本发明涉及对总线上的各进程之间的工作流程的控制，特别涉及对分享处理器资源的各进程间的逻辑连接的管理。

在以前的进程到进程的通讯中（通讯方法对于通讯进程是透明的），一个包含进程的单元请求总线上的另一单元来完成工作。待操作的数据放在请求者的存储中，而服务者单元，亦即执行此请求的单元，有存取这些数据的方法。以前的系统缺乏简便管理进程间的逻辑连接以便可用最低级别服务的手段。服务级别依赖于所分配的系统资源。

从对进程编程的观点来看，所分享的资源是由服务者进程提供的工作。该资源的各单元是那些工作请求。必须保证任何服务器的所有用户具有存取此资源的最低级别，亦即，必须允许它们具有若干个未完成的工作请求，且保证这些请求最终将提交给服务器进行处理。

大多数的总线执行使用独立的信号线来确认接受还是拒绝表明工作请求的信息。对于远程通讯应用，对流程控制已实现了步调机构（pacing mechanism）。SNA使用一种计数机构，允许信息启动器在接收到一个确认信号以前有最大数量的待完成信息。可以处于待完成状态的信息数目必须由信息接收器建立，且待完成信息的步调控制必须由信息发送器负责。这样，流程控制中直接涉及每个独立的发送器和接收器。

在 Robinson 等人的美国专利 No. 4, 449, 182 中, 在主处理器的主存储中储存了一个指令环与一个响应环。它们包含对 I/O 控制器处理器的指令的空间和响应来自控制器的指令的空间。对每一个 I/O 控制器有一个响应环和一个指令环。使用一个位来表明所有权以防止某个处理器写一条还没有由 I/O 控制器处理的指令或还没有由主处理器读出的响应。若指令环满了, 或者响应环空了, 那末在主处理器接到表明指令环已经变成非满, 或响应环已经变成非空的中断以前, 它不再对环进行检查。两个处理器均含有指向环中应该读或写的下一个位置的指针。这样, 以一种十分原始的方式在各装置之间建立了一种简单形式的流程控制。

Robinson 等人的专利没有说明多进程互相通讯的资源分配问题。它不保证在任何一个处理器中运行多个进程可提供最低级别的服务。在多进程环境中的通讯由于给定的不同排队延迟和进程的独立性而变得复杂得多了。

一种分布式处理器网络中的总线单元通讯管理机构, 管理通讯进程之间的逻辑连接以保证对每个进程提供最低级别的服务。总线单元是隶属于分布式处理器网络中的总线的处理单元, 且可以包含一个或几个主处理器或者输入/输出处理器。进程之间的逻辑连接由总线管理归总为在每个处理器处的连接群。每个连接群具有足够的处理器资源来完成至少一个工作请求。当一个连接群已达到它的待完成的工作请求上限时, 总线管理将该群已充满后所请求的工作通知包含这些进程的总线单元, 并表明此时不能完成它们指定的请求。

由于硬件排队的延迟, 请求进程可以发送一个给连接于已满的连接群的进程用的进一步的工作请求。总线管理将再一次拒绝该请求。

并通知发出请求的总线单元。当连接群中的进程完成工作后，总线管理将通知该总线单元它现在可以接受进一步的工作请求了。由于希望按发送的次序来处理工作请求，所以该请求总线单元将按正确的次序重新开始它的工作请求，然后通知服务进程的总线单元它应该再开始接受工作请求。只有在从请求程序接到重新开始的信息以后，请求总线单元才发送将由服务总线单元接受的请求。

使用连接群具有明显的好处。这是一个在最低编程级别上的单一流程控制机构，且用于所有的总线单元。总线单元的独立任务、主处理器，或者支持诸如通讯机构、二级存储器、工作站等不同形式装置的I/O控制器，都使用该流程控制机构。由于它对所有总线单元是相同的，所以它还在总线上提供一种相互处于平等地位的环境。

这些连接群防止任何总线单元由于繁忙活动而被其它总线单元切断另一个总线单元的服务。通过对每个隶属于总线的处理器请求至少一个连接群，且使该连接群仅用于两个特有的处理器上的进程之间的连接，来做到这一点。这样，一个耦合于其它两个处理器的处理器必须至少有两个连接群，每个处理器有一个。

在一个较佳实施例中，有三种类型的连接群用于与其它要求通讯的处理器中的每一个进行通讯。活动不多的进程属于一个连接群，它在任何一个时间只能接受不多几个工作请求。中等活动的进程属于一个连接群，它可以接受更多的工作请求。活动繁忙的进程属于一个连接群，它具有很多系统资源以接受大量的工作请求。这样，基于期望的工作级别，对不同的进程保证相应的服务级别。

使用上述信息来指明何时一个连接群是满的以及调整重新发送在那个连接群中曾被拒绝的信息，这些提供了更多的好处。它保证工作

以所希望的次序来完成。

各连接群允许对其作用是释放处于关键状态的资源的进程指定一个连接群。这样，若该连接群可以接受至少一个工作请求，则某些资源可以得到释放。释放资源的一个方法是关闭整个工作并重新使该系统置位。

对资源分群的进一步的好处是提供支持更多装置的能力。每个装置具有它运行时所需要的最小量的资源。在一个连接群中提供的资源（每个装置的最小要求乘以该群中的装置数）越少，可以隶属于一个系统的装置就越多。一个例子是一台磁带驱动装置。若要求它隶属于多个磁带驱动器，而每一个驱动器需要大量的存储用于缓冲数据，则对每一个磁带驱动器指定足够的资源将会只留下很少的可用于直接存取存储装置的资源。将这些磁带驱动器组合为一个连接群，并且仅为一个或两个同时运行的驱动器分配足够的资源，则有足够的资源可用于直接存取存储装置。对磁带驱动装置，并且也对直接存取存储装置提供有保证的支持级别。由于不大可能会五台磁带装置同时运行，所以没有降低总的服务。

图1是取自美国专利 N o. 4, 649, 473 的方块图，它是具有用于各进程之间通讯的进程间通讯设备的多进程系统的总图。

图2是图1所示的多进程系统的总体方块图，它给出在各进程间有逻辑连接的逻辑连接群。

图3给出了用于管理图2中的逻辑连接群的表和控制块。

图4是表示一个请求总线单元的方块流程图，该单元对指定给一个连接群的工作请求的“队列满”（Queue Full）状态作出响应，该连接群没有足够的资源来接受更多的工作请求。

图 5 是图 4 所示的对“队列满”状态作出响应的请求总线单元的进一步的流程图。

图 6 是由图 4 所示的对“队列满”状态作出响应的请求总线单元使用的队列的方块图。

图 7 是服务总线单元的方块流程图，表示确定连接群中是否具有足够的资源以及如果没有足够的资源就发送“队列满”信息的流程图。

图 8 是图 7 所示的服务总线单元的进一步的流程图，它决定何时连接群的资源已经成为可用的，并发送一个“队列空间可用”（Queue Space Available）信息。

图 9 是一个方块图，表示报告各种错误和指明“队列满”状态的“总线错误条件”（Bus Error Condition）信息的各字段。

图 10 是一个方块图，它给出了表明该资源对连接群是可用的“队列空间可用”信息的各字段。

图 11 是一个方块图，它给出了通知连接群开始接受工作请求的“重新开始排队”（Restart Queue）信息的各字段。

图 12 是包含连接群的队列满条件的信息流程图。

图 13 是一个方块图，它给出了表明存在一个对确定进程的工作请求的“运行开始”（Opstart）信息的各字段。

图 14 是一个方块图，它给出了用来确定各请求和数据的位置的“请求/响应控制块”（Request/Response Control Block）的各字段。

图 15 是表示附加的“请求/响应控制块”各字段的方块图。

图 16 是一个方块图，它给出了表明存在一个对工作请求的响应

的“运行结束”(Open)信息的各字段。

图17是一个方块图,它给出了请求管理远程存储的“存储请求”(Storage Request)信息的各字段。

图18是一个方块图,它给出了表明待远程管理的存储表的位置的“存储表可用”(Storage List Available)信息的各字段。

图19是一个方块图,它给出了用来确定待管理的远程存储的“存储表控制块”(Storage List Control Block)的各字段。

图20是一个方块图,它给出了用来返回远程存储管理的“存储表完成”(Storage List Complete)信息的各字段。

图21是一个方块图,它给出了用来请求返回远程存储管理的“返回存储表”(Return Storage List)信息的各字段。

图22是传送远程存储管理控制的信息流程图。

图23是表示准许远程存储管理的方块流程图。

图24是表示返回远程存储管理的方块流程图。

图25是逆转工作请求流程的信息流程图。

图26是一个方块图,它给出了用来非正式传送少量数据的“信号”(Signal)总线信息的各字段。

图27是一个方块图,它给出了图26所示的用于逆转工作请求流程的“信号”信息的不同版本。

图28是一个方块图,它给出了用来请求DMA传送的“DMA请求”(DMA Reg)总线信息的各字段。

图29是一个方块图,它给出了用来表明DMA传送已完成的

“DMA完成”(DMA Comp)总线信息的各字段。

图30是使用图28和图29的DMA信息传送数据的信息流程图。

图1取自转让给本发明的同一受让人的美国专利NO.

4,649,473,它用来向读者提供,更清楚地理解本发明的基础。紧跟着的内容叙述图1,并且集中于进程到进程的通讯。这里这些进程感觉不到潜在的通讯管理。将在稍后一节描述该潜在的总线传输机构,以及乃是本发明主题的总线管理。

在图1中10表示一个分布式进程环境的示意图。由12代表的处理器A通过由线14代表的实际路径耦合到由16代表的处理器B。处理器A具有在其内部的由18代表的进程A以及由19代表的进程B。存储区域20与进程A以及进程B相连接,提供数据存储的进程控制和存取,这种连接分别由线21和22表示。

处理器B具有在其内部的由23代表的进程C以及由24代表的进程D。存储区域25与进程C以及进程D相连接,提供数据存储的进程控制和存取,这种连接分别由线26和28表示。

在这些处理器中,各进程或执行程序需要互相通讯。在不同结构的处理器中,或者在同一处理器中随着时间的改变,进行通讯的两个进程可以处于不同的相对位置,它们之间可以有不同的实际路径。

在处理器A和处理器B中提供了分别由30及32代表的进程间通讯设备(IPCF),它调节对通讯进程是透明的进程间通讯。

IPCF 30通过线34与处理器A中的进程A相耦合,通过线36与进程B相耦合。线34和36代表进程A和进程B到IPCF 30之间的接口。只要建立了适当的数据通道,这些接口就允许在进

程A和进程B之间进行通讯。IPCF 3.0还通过处理器B中的传输机构40与IPCF 3.2相耦合。IPCF 3.2本身通过接口线42和44与进程C以及进程D相耦合。这些IPCF的接口和传输机构允许在上述所有进程之间建立通讯，而不需要关于与之通讯的进程的位置的进程知识。传输机构38和40最好包含多个传输机构，诸如在单一处理器中进程A和进程B之间或进程C和进程D之间进行通讯时使用的局部传输结构。若处理器A和处理器B在同一个机器中，则总线传输机构被用来进行处理器A和处理器B的各进程之间的通讯。对于机器之间的通讯，诸如SNA的通讯协议是适用的。

传输机构38、40是数据传送器。它们负责把数据的名字节从一个地方传送到另一地方，而并不理解所传送的信息的意义。这样，线46代表处理器A中的存储20和传输机构38耦合，线48代表处理器B中的存储25和传输机构40耦合，以允许传输机构38、40在存储器20与25之间直接传送信息。

IPCF为企图进行通讯的进程选择进行该通讯的传输机构。进行通讯的进程不需要知道所用的机构。企图进行通讯的进程将目标进程的名称提供给IPCF（这对企图进行通讯的进程而言是已知的）。IPCF用一个适当的目录服务来安置它。然后IPCF选择适当的传输机构并使用系统提供的服务，以标准方式建立进程间的连接。所有级别的进程，从应用性的到诸如分页管理那样的基本系统服务，都可以使用IPCF。

为了允许使用各种不同的具有不同能力与特征的传输机构，IPCF包含一个对每个进程的通用传输机构接口。该接口确定了一组为建立连接以及在进程间传递信息的功能。这些功能被变换到

I P C F所使用的传输机构上。写到接口上的程序是独立于传输机构的，因而是独立于它们在通讯时的相对位置的。

进程间的通讯是通过由I P C F建立它们之间的连接来发送和接收信息而实现的。该信息包含工作请求和/或数据。相对于一个特定的工作请求，进程起着请求者或服务器的作用。请求者通过发送请求给执行此请求的服务器而启动工作请求。请求包含工作请求（指令和它的参数）以及可选的某些数据。请求和数据均具有可变的长度。

如果读者需要关于I P C F的进一步的信息，以及建立连接的方式，可参考转让给本发明同一受让人的美国专利NO. 6, 649, 473。现在描述作为传输机构38和40一部分的总线管理。

总线管理

在图2中，两个总线单元50和52通过一个诸如I/O（输入和输出）总线54那样的实际路径相耦合。在一个较佳实施例中，总线单元50包含一个带有处理器56的主处理系统，主存储器58通过线60与该处理器相耦合。主处理器56执行多个程序，从计数程序到操作系统程序。一个程序的一次执行称为一个进程。在主处理器56中给出了几个进程PA1-PA_n，PB1-PB_n，PC1-PC_n。

在一个较佳实施例中，总线单元52包含一个I/O处理器66，主存储器68通过线70与其相耦合。I/O处理器66也有许多可执行的进程PD1-PD_n，PE1-PE_n以及PF1-PF_n。在处理器56和66中的每一个中画出了大量的进程，但也可能出现一个处理器只有一个进程的情况。在另一个实施例中，处理器66是一个处理器网络中的一个普通的处理器。进程间相互进行通讯，而不

管通过总线单元50的IPCF72和通过总线单元52的IPCF74的两个通讯进程的位置。每一对通讯进程是通过IPCF由总线单元50中的线80和总线单元52中的82所表示的逻辑连接在逻辑上连接起来的。

在前述专利No. 6, 649, 473中更详细地描述了这些连接的建立。实质上, IPCF的“打开”(Open)谓词在两个进程间建立了IPCF连接。对于每一对进程的通讯, 该连接是唯一的。这样, 线80中的每一条与线82中的一条相匹配。因为可以有比图2中画出的更多的I/O处理器或主处理器, 所以比线82更多的线80指明对没有画出的处理器中的进程的连接。由一个要求开始通讯的进程发出“打开”谓词, 使得IPCF在发出“打开”谓词的进程和由“打开”谓词确定的目标进程之间建立逻辑连接, “打开”谓词的目标由一个实体名称确定。“打开”谓词基于提供给它的实体名称建立一个程序的新的或已经执行的实例的连接。

“打开”谓词包含IPCF使用的实体名称和相应的操作系统, 以便确定服务器中需要连接的程序以及此程序的执行例(即进程)。一个连接识别符确认由IPCF返回的连接。它用来参考在相继操作中的这个连接。一个特定的连接识别符只在单一处理器中有效。一般说来两个相连接的进程对于同样的连接有不同的连接识别符。连接识别符由局部IPCF指定, 并且在一个处理器中是唯一的。IPCF使用返回码来表明“打开”谓词的进程已完成。

总线管理86被表示成通过许多已经建立起来的逻辑连接80与主处理器56相耦合。在总线单元52中画出一个类似的总线管理88, 它也具有多个连接82。总线管理86提供在图1所示总线单

元中的总线传输机构 (BTM) 38 和 40 所完成的同样功能。

总线管理 86 和 88 管理这些连接, 并且发出信息来控制总线 54 上的工作请求的流程。标以 90 和 92 的支持此总线的硬件, 在实现总线控制允许传送信息和数据以前, 提供每一个总线单元任意和总线相接的能力, 提供到另一个总线单元的主存储器的 DMA 能力, 并且对来自各个总线管理的信息进行排队。总线硬件还控制从主存储器队列中的总线单元来的信息的排队。

总线管理 86 接收由总线单元 52 发出的“打开”谓词所产生的通讯。然后它建立与所指定的进程的连接。利用关于诸如目标进程 (即服务进程) 的特性, 请求进程的特性, 特定装置所完成的工作的类型, 或其它因素的信息, 把上述连接的连接识别符赋予标号为 CG1, CG2, ..., CGn 的许多连接群中的一个。该信息中的某一些包含在与总线单元 52 的通讯中。另一些信息包含在主存储器 58 中, 并且根据总线单元 52 提供的信息进行存取。然后根据那些信息进行指定, 并且在一个较佳实施例中是由开发者预先构造的。

连接群用于控制工作请求的流程。每一个工作请求要求处理一定量的系统资源。若在先来先服务基础上安排对所有工作请求的服务, 那末许多高资源要求的装置将使其它装置不能得到服务。

要求大量主存储资源的装置的一个例子是磁带驱动装置。为了保持磁带装置以它最有效的流水模式运行, 要求一个很大的主存储缓冲器。如果希望连接多个磁带装置, 并且每一个均要求大量的存储来缓冲数据 (当它以流水模式运行时), 则对每一个磁带驱动器指定足够的资源将使可分配给直接存取存储装置的资源留下很少。通过将一些磁带驱动器组合为一个连接群, 并且仅为同时运行的一个或两个驱动器分配足

够的资源，则有更多得多的资源可用于直接存取存储装置。对磁带驱动装置，并且也对直接存取存储装置，提供有保证的支持级别。由于不大可能会五台磁带装置同时运行，所以没有降低总的服务。

连接群还可以用来保证总有可服务于某些主进程的資源。一个例子可以是把主保存/存贮进程和相应的 I/O 进程之间的服务器一侧的连接放进一个连接群，在那个群中它是仅有的连接。这样，在服务器进程可以与其它群连接的同时，总是有资源可服务于主保存/存贮工作请求。

分配群的一个基础是装置的运行特征。各种数据传送器，诸如磁带驱动器，它们只是偶尔用到的，放在一个群中。诸如直接存取存储装置那样的装置，它们繁忙地用于将数据存入主存储器以及从主存储器中取出数据，放在另一个群中，并且给这个群分配大量的资源以保证高级别的服务。其它装置，诸如工作站，放在又一个群中。若存在足够的可用资源，则每个装置有它自己的群。

当讨论把装置放入群的时候，作为一种简便说法，我们说把联系于该装置的进程的连接放进了一个群。根据所服务的工作请求的类型将各个连接进一步分配给各个群。与错误条件处理有关的连接以及与其它维修有关的工作请求可以放入它本身组成的又一个群中，以保证对这些请求的最低级别的服务。可以看出，由于使用连接群来分配资源和这样控制工作流程，而提供了极大的灵活性。

在作为服务器工作的总线单元 50 中，每个连接群均列在图 3（其数号与图 2 相一致）的主存储器 58 中的控制块 120 中。控制块 120 包含每个连接群 CG 1 - n 的项目，用于群识别符的标以 GPID。在 122 栏中表明的数是与 GPID 相联系的项目，它用

来保持跟踪通过该群中的一个连接而收到的工作请求数,但是该请求还没有被总线单元50中的进程服务过。活动连接识别符(OID)的表125有指向连接群项目120的指针,并且确定有一个连接是其一部分的那个群。总线单元52有保持跟踪等待的尚未服务过的工作请求数的类似机构。

上述工作请求流程的好处之一是,连接群中的工作(对于具有同样优先级的单一连接)是以请求总线单元所要求的次序得到服务的。这样,如果对一个进程有多个某优先级的写入数据请求,那末将按要求的次序写入,即使有某个时刻资源不够用也是这样。所有这些对于请求进程是明显的。

现在参考图4和图5来描述请求总线单元中的流程图。使用4个队列来保持跟踪发源于请求总线单元中的工作请求。每一个队列包含指向工作请求的指针,请求识别符以及在“打开”进程期间与其通讯的那个服务总线单元中的相应的连接群。“准备发送队列”用在正常的工作请求流程中,以指向准备发送给服务处理器的工作请求队列。当为“准备发送队列”中的一个请求发送一个“运行开始”(OP START)信息时,移掉该指针并将它放入“等待响应队列”。当收到一个“运行结束”(OP END)信息时,从所有队列中移掉该工作请求的项目。在图4中由方块150、152和154表示了此正常流程。

当根据一个工作请求返回“队列满”时,该指针从等待完成队列移到由156表示的拒绝队列,并且表明收到了“队列满”信息的标志是与收到此信息的连接群相联系的。然后在158处检查表明现在该连接群有空间可用的“队列空间标记”(Q SPACE FLAG)。

正常状况下该标记为 off，因而总线管理继续发送和接收通讯。以这种方式把在等待完成队列中的、“队列满”标记被返回的每一个工作请求放入拒绝队列。

在图 5 中，当总线管理准备好发送与“准备发送队列”中下一个项目对应的另一个“运行开始”时，它在 160 处对服务总线单元中的连接群检查“连接群满”标志。若它表明该连接群队列是满的，则该请求项目被放入 162 处的中间队列。若它不满，则在 164 处发送一个“运行开始”信息，并且在 166 处将该请求项目放入等待完成队列。

回到图 4 所示的流程，当请求处理器在 168 处收到一个“队列空间可用”（Queue Space Available）信息时，在 170 处设置“队列空间标记”ON，并且在指定连接群中对位于收到“队列满”项目之后的任何项目，检查等待完成队列。若对所有等待完成的请求“队列满”均没有被返回，那末总线单元在 174 处继续等待下一个通讯。

如果在 172 处全都得到了确认，则在 176 处把这些项目从拒绝队列移到准备发送队列的开始处。然后，在 178 处把请求项目从中间队列移到准备发送队列，放在从拒绝队列来的项目之后，并在 180 处重新设置“队列空间标记”。在 182 处发送“重新开始排队”（Restart Queue）信息，并且总线管理在 184 处开始从准备发送队列发送。

在图 6 中，画出了上述流程所用的队列。箭头表示相应于工作请求的队列项目的流动。正常情况下，准备发送队列 200 经过线 202 把项目传送给等待响应队列 210。如果收到了“队列满”状

态，从队列 2 1 0 经过线 2 2 2 把项目传送给拒绝队列 2 2 0。同时，当新的工作在队列 2 0 0 中开始时，如果检测到对一个已经满了的连接群有工作请求，则将它经过线 2 3 2 传送给中间队列 2 3 0。当所有的项目从满连接群用的等待响应队列 2 1 0 传送到拒绝队列 2 2 0，并且根据“队列空间可用”信息已经发出“重新开始排队”信息时，通过线 2 3 4 把曾被拒绝的请求送回到准备发送队列 2 0 0 的顶端。然后，经过线 2 3 6，为现在已有资源可用于进一步工作的群准备的另一些请求被传送给准备发送队列 2 0 0，放在上述曾被拒绝的请求之后。

图 7 与图 8 中画出了服务总线单元中相应的流程。服务总线单元在 2 5 0 处接收一个“开始运行”信息，在 2 5 2 处确定连接群，在 2 5 4 处增加它的计数。使用一个计数限来确定连接群在任一时刻可以待完成的工作请求的数目。在 2 5 6 处将上述计数的数值与计数限相比较，若此值大于计数限，则在 2 5 8 处设置“队列满标记”（Q FULL FLAG），并且在 2 6 0 处向发采信号的总线单元发送在“总线错误条件信息”中的“队列满”状态，在 2 6 1 处继续进行处理。在另一个实施例中，监视实际可用的资源并与由“运行开始”表示的工作请求所要求的资源进行比较。若计数值不大于计数限，但在 2 6 2 处探测到“队列满标记”，则该流程返回到 2 6 0，并根据“运行开始”发送“队列满”状态。否则，在 2 6 4 处继续处理该请求，在 2 6 6 处继续该流程。如果在 2 5 0 处收到的信息不是“运行开始”信息，则在 2 6 8 处检查它是否是一个“重新开始排队”信息。如果不是，在 2 7 0 处继续处理。如果它是“重新开始排队”信息，在 2 7 2 处将“队列满标记”重新设置到 off 状态，且在

274处继续处理。

为了清掉队列满条件，服务处理器必须或者分配更多的资源并增加待完成请求的计数限，或者完成一个满连接中的至少一个工作请求。图8中表示了这一流程。当一个工作请求完成后，在280处减少相关的连接群的计数。在282处或者发送“运行结束”，或者发送一个“总线错误条件”信息。然后，在284处检查“队列满标记”，如果它是off，在285处继续进行处理。如果“队列满标记”已置为on，则在286处检查该连接群的计数，看它是否低于计数限。在方块286中称为“低值”的此计数限不需要与前面讨论过的计数限相同。可以期望建立一个更低的计数限以保证有足够的资源可用于许多请求。由于多个请求可以跟着请求队列的重新开始而快速发送出去，因而可以帮助消除“队列满”流的不变的序列。如果计数值不低于该“低值”，则在287处继续处理。如果在286处计数值低于该“低值”，则在288处发送“队列空间可用”信息，且在290处继续处理。

在一个较佳实施例中，在总线单元50中，为每一个通讯协议定义了分开的连接群。这些协议包括与完成X.25, LAN, SDLC及其它通讯协议的进程的连接。

另一个群包括错误处理进程的连接。另一个群用于与一般维修的进程有关的连接。

还有一个群包括常规功能，诸如与在总线单元52控制的磁盘驱动装置间传送数据的进程的连接。

连接群数目的几种变化是在本发明范围内的。某些进程可能要求大量的资源，因而与每个这种进程的连接可以要求有一个单一的连接

群。对所需连接的具体分群取决于实际执行和要互相通讯的各进程的特征。

在主处理器 56 中的一个进程与 I/O 处理器 66 中的一个进程之间建立连接，导致总线管理 88 完成该连接，并且将它的进程连接分配给多个连接群 CG 1、CG 2、...、CG n 中的一个。这样，对于包括该总线单元的系统中的每个进程，它与另一个进程有连接，分配目标进程的处理器 的总线管理分配它与一个群相连接，并且给该连接群分配资源，用来分配给该群 的连接上的工作请求服务。

流程控制的总线信息

当一个连接群没有足够的资源来为连接于该群 的请求提供服务时，信息在控制和“重新同步”信息的总线管理之间传送。当目标总线单元不能接受一个特定的连接群 的新工作时，它返回具有指定的

“队列满”状态的信息。此状态包含在“总线错误条件”信息中，它的格式示于图 9 中。“总线错误条件”信息返回到对请求正常响应的 位置来报告失败（阻止成功地完成或继续运行）。“总线错误条件”信息发送的一些其它条件包括内存存取请求中的寻址错误，格式错误，以及收到未定义或不支持的信息。另外一些条件也可以引起发送这种依赖于总线传输机构的实际执行的信息。

在“总线错误条件”信息中的字段“保留”（RESERVED）被保留在将来使用。字段“总线单元”（BUS UNIT）用来确定信息源。对本发明的目的而言，它不起作用。第三个字段，“信息识别符”（MESSAGE ID）（82）用来确定该信息是“总线错误条件”信息。“控制字段”（CONTROL FIELD）包含确定经受过失败的特有的请求或连接的信息。此字段的内容取决于该失败、该特定

的传输信息/DMA以及经受该失败的流动方法。OFID字段确定“控制字段”的内容。不同的值表明控制字段中的信息是请求识别符，或是服务连接识别符，或是控制块的地址等等。这取决于发送信息的原因以及是谁发送的。ACTN字段确定是否要进行恢复动作。一个值表示不进行任何动作，另一个值要求启动关闭一个连接。又一个值引起重新设置使通讯重新同步。

在“错误状态”(ERROR STATUS)字段中指明“队列满”条件。“连接群识别符”(CONN GROUP ID)紧跟着“队列满”条件指明那些已经满了的连接群。它表明所指定的总线单元没有执行此信息。当出现“队列满”状态时，“控制字段”将含有不同的值。它可能包含一个控制块地址或者一个请求识别符。这依赖于传送数据的方式。传送数据的不同方式将在本申请的流程控制部分中作进一步的描述。“错误状态”字段还用来确定诸如上述那些其它错误条件的状态。

在向源总线单元发送了“队列满”状态之后，发送此信息的目标总线单元的总线管理监控着相应的连接群，以决定何时有足够的资源可用来服务于该连接群。

当目标总线单元在该特定的连接群中有可用空间时，它向源总线单元发送一个“队列空间可用”信息。该“队列空间可用”总线单元信息用来向源总线管理表明队列空间是可用的。仅在总线单元向源总线单元发送了“队列满”状态以后，才由该总线单元发送此信息。它表明该连接群有队列空间可用了，图10中画出了“队列空间可用”信息的格式。有4个“保留”字段，一个“信息识别符”(MESSAGE ID)字段和一个唯一地定义了有队列空间可用的连接群的“群”

(GROUP) 字段。

源总线单元将接收该“队列空间可用”信息，并确定送到目标总线单元上那个特定连接群去的工作请求的数目。由于通讯是异步进行的，并且在源总线单元的硬件中可能存在内部排队延迟，所以源总线单元可能已经发送了更多的信息去启动工作。

当确定了要重新发送的通讯，亦即在使“队列满”状态返回信息以后产生的那些通讯，并且已指明该连接群是满的，这时源总线单元的总线管理发出一个“重新开始排队”信息。此信息用来保证在目标处理器中由连接群按次序地取出信息。如果连接群在发出“队列空间可用”信息后立即开始取信息，有可能使某些工作进行处理的次序混乱。这将使源总线单元先于其接收到“队列满”状态发出信息来启动工作，而目标总线单元管理则在发出“队列空间可用”信息以后才收到它。

图 1 1 中画出了“重新开始排队”信息的格式。它类似于“队列空间可用”信息。其中在“信息识别符”(MESSAGE ID) (OD) 字段中确定了信息的类型，在“群”字段中确定了重新开始排队的连接群。总线管理对每个开始工作的信息返回“队列满”状态，直到目标总线单元的总线管理收到“重新开始排队”的信息。

信息流程的例子

图 1 2 中画出了对目标总线单元中的连接群的队列满作出响应时的信息交换图。目标总线单元标为服务器，源总线单元标为请求器。在图上的服务器一侧画出了由服务器发送与接收的信息，因为流程是在那里进行控制的。请求器与服务器之间的箭头指向接收每一信息的总线单元。以数字次序表示信息的序列。现在按图 1 2 中出现的次序

来列出这些事件。

1. 请求处理器向服务处理器发送一个开始工作的信息，称之为“运行开始(1)”信息(在本说明书中稍后再定义)。

2. 服务处理器识别出用“运行开始”信息的连接群处于“队列满”条件，并且返回带有“队列满”状态的“错误”信息。

3. 因为总线单元的异步性质，请求器还不知道“队列满”状态，它向服务器发送第二个信息，“运行开始(2)”。

4. 服务器已经完成了以前的某个工作请求，并向请求器发出一个“运行结束”信息来表明这一点。

5. 因为服务器已经完成了以前的某个工作请求，服务器有了可用的资源或可用的队列空间，将向请求器发出一个“队列空间可用”信息。

6. 由于请求器中的硬件排队延迟，请求器还没有识别“队列满”条件，将发出“运行开始(3)”信息。

7. 服务器必须继续向请求器返回“队列满”状态，直到服务器收到一个“重新开始排队”信息。因此它返回“队列满(2)”状态，作为对识别“运行开始(2)”信息的响应。

8. 现在服务器识别“运行开始(3)”信息，并返回一个“队列满(3)”信息。

9. 现在请求器识别对应于“运行开始(1)、(2)、(3)”的“队列满”状态以及“队列空间可用”信息，并在决定了哪些信息必须重新发送和发送它们的次序之后，发出一个“重新开始排队”信息。

10. 11和12. 请求处理器以正确的次序重新发送“运行开

始”信息。

当工作已由服务器完成时，向请求总线单元发出一个“运行结束”信息以表明该工作已经完成。它的格式在以后描述。

数据流控

在图2所示的较佳实施例中，总线单元50及52中的总线硬件90及92是标明具有直接存储存取(DMA)能力的。这种能力是当今存在的大多数硬件的一种标准的硬件能力。主DMA能力允许具有它的总线单元直接对具有从属DMA能力的总线单元的主存储器进行存取，而不中断从属DMA总线单元的处理器。因此，不描述它运行的细节，因为这对充分理解本发明是不必要的。

总线单元50有一个从属DMA能力90。在该较佳实施例中，总线单元50是主处理器。从属DMA能力允许一个总线单元让其它总线单元存取它的主存储器58，而不中断主处理器56。这样，使主处理器56对主存储器58进行存取的线60，也与总线管理86和从属DMA硬件相连，允许90直接对主存储器58进行存取。接下来，这允许另一个总线单元，诸如具有主DMA硬件能力92的总线单元52，对总线单元50的主存58进行存取，而不中断主处理器56。只具有从属DMA能力的总线单元不能对任何其它总线单元的主存直接存取。同时，只具有主DMA能力的单元使任何其它总线单元都不能直接存取它的主存。

如果总线单元52中的一个进程向总线单元50中的一个进程发出一个工作请求，真实的数据传送对这些进程而言必须是透明的。

IPCF 72和74是进程用来处理工作的谓词接口。服务器进程将以其本身的步调存取由请求器确定的数据。由于这种情况下服务器总

线单元只具有从属DMA能力。所以提供一种得到对IPCF和进程均透明的数据的手段。

在保证每一个总线单元都有完整的DMA能力的标准流程中，总线管理88接受从IPCF谓词74采的信息，它表明有一个进程需要发送工作请求。然后，总线管理88通过发送一个“运行开始”信息来通知总线管理86有工作要做。图13中画出了“运行开始”总线信息的格式。此格式中有足够的信息提供给服务总线单元50中的总线管理86，使得在可能的情况下，把一个指定控制信息和数据地址的“请求/响应控制块”(RRCB)从总线单元52的主存储器68移到总线单元50的主存储器58中。图14和15中画出了RRCB的进一步的细节。然后，总线管理86可以通过IPCF72通知主处理器56中被看中的进程：有一个由该进程排队的待完成的工作请求。该进程将执行该请求。这需要在总线单元间移动数据。总线管理86将使用现在处于主存储器58中的RRCB的副本，来控制总线单元之间的数据传送。向总线管理88发出的一个“运行结束”(看图16)信息表明所请求的操作已完成。此信息通过IPCF74通知给请求进程。

如图2中实现的上述方案的问题在于总线硬件90不能对主存68直接存取。即使总线硬件90有主DMA能力，总线硬件也还必须同时有从属DMA能力。通过使用一个存储表控制块和几个新总线信息来解决此问题，以管理主总线单元50的主存58中的缓冲器，为总线单元52的总线管理88提供缓冲。这允许请求器中的总线管理88，根据工作的标准流程，把与该请求有关的数据传送给服务器主存58中的缓冲器。然后，服务器把这些数据从缓冲器传送到服务进

程可用的存储器中。这样，数据流程对 I P C F 7 2 而言是标准的。象通常一样，用 R R C B 指明服务器必须存取的数据在哪里。请求总线管理 8 8 简单地保证把数据放在主存 5 8 的缓冲器中。现在给出关于 R R C B 和信息的更详细的说明。

图 1 4 和 1 5 中画出了一个 R R C B。它用来确定一个工作请求和与它有关的数据。该 R R C B 是一个控制块，被请求总线单元的总线管理和服务器总线单元的总线管理用来控制总线单元之间的数据移动。R R C B 中的信息被用于物理的 D M A 进程。R R C B 的内容最好是只读的。该内容不能由服务器总线单元的总线管理改变或修改。

在较佳实施例中，R R C B 可以由请求总线管理指定为直至 4 0 8 8 字节的任何长度。为了利用请求器提供的固定块缓冲器的管理，可以把 R R C B 分段并用链连接起来。例如，若一个固定块的长度是 5 1 2 字节，而一个 R R C B 更长一些，则将 R R C B 分段为一个包含某些头标信息的第一类型段，和多个图 1 5 所示的第二类型段。没有一个段比固定块更长。R R C B 的第一类型段的第一个字段是总的 R R C B 的字节长度。此长度是 R R C B 的所有各段的长度的总和。一个“R R C B 类型” (R R C B T Y P E) 字段指明这是一个第一类型的 R R C B 段。“服务器连接识别符” (S E R V E R C O N N E C T I O N I D) 为该请求指明目标进程的识别符。一个“请求优先级” (R E Q U E S T P R I O R I T Y) 字段指明，当在服务进程的输入队列中插进一个请求项时，该服务处理器使用的优先级。一个“标记” (F L A G S) 字段规定服务器是需要一个确定的响应，还是只需要一个表示异常的响应。一个“请求识别符” (R E Q U E S T O R R I D) 字段指明该请求的识别符。只有该请求器知道它。

“扩展状态指针” (EXTENDED STATUS POINTER) 指明一个可以放入扩展状态 (超过状态允许的结构尺寸的状态数据) 的区域地址。这个区域必须是可用的, 并在使用前由请求总线管理置为零。该地址是在请求器管理的存储器中, 与 R R C B 的存储器一样。

第一类型段的其余部分与第二类型段相同。它包含多个指定数据类型的描述符单元字, 这些数据由**“数据标记” (DATA FLAGS)** 字段中的描述符单元所描述。在**“数据标记”** 字段中的描述符类型确定了该描述符的类型, 诸如**“请求” (Request)**, 从服务器存储到请求器存储器**“数据输入” (Data In)**, 从请求器存储到服务器存储的**“数据输出” (Data Out)**, 或者在需要另一段时是连到下一个 R R C B 段的一个段。如果存在另一个 R R C B 段, 该 R R C B 段连接描述符单元必须放在该 R R C B 段的末尾处。在**“数据标记”** 字段中的描述符格式字段用来指明立即数在数据字中有正确的起点, 并延续至最大 44 个字节。一个请求或**“数据输出”** 描述符可以是立即数或者一个参考量, 该参考量包括总线单元号和确定 D M A 数据来去地方的数据地址。在指定该参考量的描述符格式时必须有总线单元号。**“数据标记”** 字段由总线单元号来确定下一个字段中的地址涉及的总线单元。

“数据长度” (DATA LENGTH) 字段指明由紧跟着的地址字段指定的字段的数据的字节长度。它是一个无符号的整数, 指明邻接的实际存储量。以零来填补此数据使之成为 8 字节的倍数。

“数据地址/数据” (DATA ADDRESS/DATA) 字段用作一个地址或立即数。如果在 R R C B 段的前述**“数据标记”** 描述符格式

中指定了立即数。这个字段就是个立即数；否则它是一个地址。该地址可以在服务器的存储中，在请求器的存储中，或者在第三个没有画出的总线单元的存储中。服务总线管理使用此地址进行来自或去到一个存储器的DMA操作。该存储器可以在其它处理器中或者在服务器存储中由请求器控制的缓冲器中。

缓冲器管理信息

在两个总线管理之间传递缓冲器管理。一个总线单元将提供在它的主存储由其它总线单元使用与管理的远程存储。在此实施例中，总线单元52对和主处理器56紧密耦合的主存58中的缓冲器进行管理。总线单元52，远程处理器66可以为任何符合其要求的目的来使用主存58中的远程存储。远程处理器可以把此远程存储看作它本身存储的一个逻辑扩展。

总线单元52通过由总线管理88发出一个“存储请求”(Storage Request)总线单元信息来请求远程存储。图17画出了“存储请求”信息的格式。标准系统一建立起来，总线单元立即使用此信息来得到远程存储。

当远程处理器没有任何进一步的缓冲器可用时，也发出此“存储请求”总线单元信息。可在“缓冲器长度”(LENGTH OF BUFFERS)字段中指定要求的缓冲器的长度。局部的处理器可能不提供所请求尺寸的缓冲器，但如果提供一个更大尺寸的缓冲器则将满足此请求。不提供更小尺寸的缓冲器。有许多“保留”字段，且被指定为零。“信息识别器(06)”字段确定此信息为“存储请求”总线单元信息。“存储尺寸”(STORAGE SIZE)字段是所请求存储的字节长度。在较佳实施例中，一个单个信息可以请求的最大存储为65535字节。“缓冲器长度”字段指定所要求的缓冲器的最小长度。这样，当要求的总的存储尺寸不能满足时，如果只提供一个缓冲器，则它至少和“缓冲器长度”字段中的值一样长。

由局部总线单元中的总线管理发送“存储表可用”(Storage

List Available) 总线单元信息和“存储表控制块”(SLCB), 响应“存储请求”总线单元信息。SLUB在局部总线单元的存储中给出一系列可由远程总线单元使用的缓冲器。只发送一个“存储表可用/SLCB”响应“存储请求”, 且只响应一个“存储请求”。

图18画出了“存储表可用”总线单元信息的格式。“标记”字段指明响应的类型, 诸如存储器是可用的, 资源不可用并且不能提供存储, 或者没有所要求那么大的缓冲器可用以及不对该请求提供任何响应。“信息识别符(07)”字段识别这是一个“存储表可用”总线单元信息。“存储表控制块的地址”(ADDRESS OF STORAGE LIST CONTROL BLOCK) 字段指定SLCB在包含远程存储的总线单元存储中的实际地址。只在“标记”字段的值表明该存储可用时, 此字段才可用。“长度”(LENGTH) 字段指明局部总线单元存储中的SLCB的长度。

图19中是SLCB的格式。“总线号”(BUS NUMBER) 字段指定局部总线单元中出现的总线单元的总线号。在较佳实施例中, 可以有多至8个不同的总线。“总线单元”字段指定该SLCB指向的远程总线单元的总线单元号。在下两个字段中指定缓冲器号和它们的长度。局部总线单元中的发送器的责任是保证此字段与“存储表可用”信息中的“长度”字段相符。

包含远程存储的总线单元中的总线管理着同一总线单元上的主存储的一部分。它保持对主存储中的缓冲器进行跟踪, 并且准许控制请求主存储的其它总线单元。这样, 连接和其它总线单元通过总线管理来争夺主存储中的缓冲器。

用“缓冲器地址”(BUFFER ADDRESS)字段来指定局部总线单元中缓冲器的实际存储地址。按要求多次重复这过程,使“缓冲器号”字段中指明的所有缓冲器均得到处理。

由远程总线单元的总线管理发出一个“存储表完成”(Storage List Complete)总线单元信息,指明远程总线单元不再要求存取SLCB指定的远程存储。还可以由具有从属DMA能力的总线单元来启动返回远程存储,它使存储可用于具有主DMA的另一个总线单元,用一个“返回存储表”(Return Storage List)总线单元信息来指明应该返回到未用的缓冲器。“存储表完成”信息还用来指明不能满足“返回存储表”中指定的请求。

图20中给出了“存储表完成”信息的格式。“标记”字段指明是正常地返回到存储表,或者是拒绝“返回”请求。它可以指明发生的是下述那一种情况,由整个表构成正常返回,响应“返回存储表”信息构成返回,在“返回存储表”信息中指明的具有指定尺寸的缓冲器构成返回,或者已找到了要求的存储,但还需要使用且不能返回。

“存储表控制块的地址”字段是局部处理器中的SLCB的实际地址。当“返回存储表”包含缓冲器的长度,且“存储表完成”信息中缓冲器不被返回时,一个“缓冲器长度”字段包含“送回存储表”中指定的各缓冲器的长度。

图21中画出了“返回存储表”信息的格式。请求器可以指定任何具有要求的缓冲器尺寸的存储表被返回,或者确定一个特定的被返回的存储表。这个总线单元信息的“标记”字段指明任何具有指定缓冲器尺寸的存储表或者一个指定的存储表是否被返回。正如上面所描

述的,这表明由“存储表完成”总线单元信息把存储表控制从远程总线单元控制交给了局部总线单元控制。“信息识别符(09)”字段确定此信息作为“返回存储表”信息。“请求的存储表的地址”

(ADDRESS OF REQUESTED STORAGE LIST) 字段指定被返回的存储表的地址,如果“标记”字段表明要求一个指定的表的话。“缓冲器长度”字段指定由于这一请求被返回的缓冲器的长度,如果没有要求指定的存储表的话。

图22中给出了带有存储表的简化的流程图。在此图的右边是具有主DMA的请求总线单元,在它下面列出了各个信息。在此图的左边是具有从属DMA的服务总线单元。下述步骤1—6描述且确定了这些操作:

1. 在具有主DMA能力的请求处理器中的总线管理通过发送一个“返回存储表”信息来通知服务处理器;它需要缓冲器。
2. 服务处理器中的总线管理发送一个信息指明有一个存储表控制块SLCB可用于请求处理器。
3. 远程总线管理将该SLCB的全部或一部分DMA到它的存储中。
4. 请求处理器中的总线管理按要求使用这些缓冲器。
5. 服务处理器中的总线管理想做某种类型的关闭,诸如一天的末尾,或者一个总线单元超过峰值负荷且没有被返回存储。它向请求处理器发送一个“返回存储表”信息。
6. 请求处理器中的总线管理发送一个“存储表完成”信息,表明不再需要该SLCB了。

在图23中,当一个仅有从属DMA能力的总线单元的总线管理

收到该总线上一个通讯时，它在300处检查此通讯是否是一个存储请求。如果是，则如块302、304和306表示的那样，该总线管理用众所周知的存储管理技术向系统存储管理请求主存储来满足该请求。该总线管理固定住这些被准许的存储，表明它有管理这些存储的责任。管理责任包括对这些存储进行读和写，并在某种级别上保证其它总线单元不独立地使用这些存储。然后，它将这些存储分块成要求的数目，建立存储表控制块，将连接于请求这些存储的总线单元的指针保持在这些被准许的块的存储地址处。然后，总线管理发送存储表可用信息给请求总线单元，并在308处继续处理。

回到方块300，如果该总线通讯不是存储请求，在310处检查此通讯是否为一个存储表完成信息。如果是，那末在312处，指明的存储被返回给系统存储管理，指向返回该存储管理的总线单元的指针被删除。在314处继续处理。如果在方块310处，没有检测到存储表完成信息，在316处继续处理。

如果只具有从属DMA的总线单元的总线管理收到一个请求，表明主处理器本身要求返回存储，则应在方块330处进入图24给出的流程。可以由于操作员的关闭指令或某种类型的定时中断产生这种请求。在任何情况下，总线管理一接到返回存储请求就检查由于被其它总线管理的管理所固定住的存储，并选择被返回的存储。该总线管理可保留某种类型的关于使用量的统计数字，或者该请求可指定要关闭的装置类型。在这种方式下，总线单元在存储中选择要求被返回的那些存储。在332处，向期望的总线单元发送一个总线信息来返回这些存储。如图23的流程中指明的那样，该总线单元将返回一个“存储表完成”信息作为应答。然后在334处继续处理。

还可用远程存储管理机构来平衡系统的品质。如果一个总线单元没有适当的主存储，它可以申请使用主处理器的主存中的缓冲器。给定了适当的总线品质，可以提高总线单元的品质。系统可以容易地跟踪其它总线单元的品质，并为那些响应工作请求的总线单元分配更多的远程存储。如果太多的主存储是远程管理的，则此分配量被主处理器品质的潜在下降所抵消。

反向流程

正如上面所讨论的，正常的工作请求流程是从主处理器 5 6 到 I/O 处理器 6 6 的。它们包含的工作可以是，向耦合于 I/O 处理器 6 6 的二级存储装置读写数据，或通过 I/O 处理器 6 6 启动通讯。对于这种关系，理想的组合是有一个具有主 DMA 能力的 I/O 处理器 6 6 和一个具有从属 DMA 能力的主处理器 5 6。I/O 处理器作为服务器不中断主处理器地进行数据传送。

往往 I/O 处理器 6 6 会有一个向主处理器发出工作请求的进程。与此工作相联系，产生了数据的反向流动。由于主处理器不能从 I/O 处理器 DMA 数据，所以 I/O 处理器利用前面讨论过的信息来使用在主处理器的主存储中得到的远程缓冲器。

现在参考图 2 5 来说明反向流程的一个例子。该图的右侧给出了一个具有主 DMA 能力和 I/O 处理器 6 6 的请求总线单元以及所完成的步骤，而该图的左侧给出了一个具有从属 DMA 能力和主处理器 5 6 的服务总线单元以及所完成的步骤。

下面说明这些标号为 1—10 的步骤。

1. 请求处理器把在进程到进程谓词接口处的一个请求发送给图 2 中 IPCE 7 4 所代表的进程到进程机构。

2. 若总线管理88在主处理器的主存58中还没有一个适当尺寸的缓冲器,它就得到一个,并通过总线硬件92启动一个主DMA操作,把该请求送进服务总线单元50中的远程存储中。

3. 请求总线管理88把数据DMA入缓冲器。

4. 然后,总线管理88把此RROB DMA入服务器存储中的缓冲器中。这时,RROB使用了该请求的服务器存储中的地址和在步骤2与3中被DMA入服务总线单元的远程存储中的那些数据。

5. 请求处理器66中的总线管理88向服务处理器发送一个“运行开始”总线单元信息,指明RROB和数据是在服务器存储中。在发出“运行开始”信息的时候,与使用进程到进程机构的进程相联系的所有数据已经被DMA入到服务器存储中。

6. 进程到进程机构72把该请求传送给服务进程。

7. 服务进程请求处于服务器的存储中的必要的数据。在处理“运行开始”信息时,总线管理88仍控制着这些缓冲器,但总线管理86也存取这些缓冲器。服务器进程的总线管理86把数据送入主存58中对服务进程能够存取的一个区域中。

8. 当服务进程完成了请求的操作时,用IPCF谓词来告诉进程到进程机构72。

9. 服务处理器56中的总线管理86发出一个带有状态信息的“运行结束”信息。这把总线管理86放入缓冲器中的任何响应都告诉了请求处理器66中的总线管理88。在收到这些响应以后,总线管理88释放这些缓冲器以备以后使用,并向请求进程表明所请求的操作已经完成。

10. 然后,进程到进程机构74通知请求进程。操作已经完

成。

用信号信息的反向流程

图 2 6 中给出了一个“信号”(Signal)总线单元信息的格式。“信号”总线单元信息由一个总线管理产生,并用来向另一处理器中的一个进程传送少量信息。在传送数据时一次使用直到 4 个字符。发送“信号”的总线管理不要求信号总线单元信息的接收者发出响应。在该请求器和一个服务进程之间的较高级别协议中也可以有一个响应,但在该较佳实施例中总线管理不要求任何东西。“信号”的发送者不能保证服务总线单元收到它了。对“信号”信息不存在流程控制机构。没有总线管理机构通知发送者“信号”无法执行,因为,例如,收到的进程没能得到存储。因为不要求响应,使它具有灵活性。和工作请求相反,使用信号信息的花费较少。不要求 R R C B。

“信号”包括两两保留区域,在较佳实施例中它们是零。

“2 X”字段的类型用来定义总线单元信息的类型。在此字段中的“2”定义这是一个“信号”总线单元信息。“X”指定“用户数据”(USER DATA)字段的内容。下面是“信号”信息的类型:

2 0 —— 注意(提醒接收者注意)

2 1 —— 立即数—1字节

2 2 —— 立即数—2字节

2 3 —— 立即数—3字节

2 4 —— 立即数—4字节

2 5 —— 立即错误数据

2 6 —— 类型 1 的立即用户数据

2 7 —— 类型 2 的立即用户数据

2 8 —— 类型 3 的立即用户数据

2 9 —— 类型 4 的立即用户数据

2 A - 2 F —— 保留给以后用

信息的“用户数据”字段包含了用户定义的数据，亦即立即数。

该字段中的立即数最好保证是正确的。“目标识别符”

(T A R G E T C I D) 字段为此总线单元信息确定目标进程。

使用另一种“信号”提供一个机构，来逆转传递工作请求和相关数据的责任，此传递过程原来是从具有主 D M A 的总线单元出发的，现在更改为由具有主 D M A 的总线单元来传送。如图 2 7 中所示，对另一种远程存储使用了“信号”一种不同形式。这是反向流程的一种容易实现的版本，但它不提供反向流程的远程存储版本所提供的那种保证。

当一个 I / O 总线单元需要向主总线单元中的一个可靠性 / 可用性 / 可维护性 (R A S) 类型的进程启动一个请求时，它发送一个具有图 2 7 中给出的格式的“信号”信息。定义“用户数据”为一个两字节的“长度” (L E N G T H) 字段和一个两字节的“偏置”

(O F F S E T) 字段，“长度”字段指明将要接收的记录的长度。

“偏置”字段是由 I / O 总线单元指定的一个编码值，且用作一个与“信号”信息有关的跟踪机构，此信息带有根据三种类型的“信号”信息之一产生的工作请求。下面定义相应于 I / O 总线单元工作请求的这些类型：

2 6 —— 类型 1 的请求 —— 错误数据

2 7 —— 类型 2 的请求 —— 资源数据

28——类型3的请求——测试数据

可以容易地确定其它的类型。

RAS型进程从I/O总线单元检索“类型请求”(Type Request)。它返回该请求中的“信号类型”、“偏置”和“长度”字段的值。I/O总线单元返回确认过的“类型请求”指令和相关的数据以响应此请求。这些“类型请求”的每个格式就好象它已经用以前的反向流程方法或正常流程发送过了。

使用这种反向流程方法的一个例子是检索错误数据。接受此“信号”的主处理器负责发送工作请求去检索指令字节和相关数据。下列工作请求字段包含给出的“信号”信息字段。

“目标”(TARGET)——“信号”类型

“地址”(ADDRESS)——缓冲器“偏置”

“最大值”(GETMAX)——“信号”长度

然后，I/O总线单元将以FIFO次序为相应于类型1，2或3请求的进程从所要求的队列之一返回下一个可用项目。

“长度”字段指定I/O总线单元将返回的数据的长度。例如，在较佳实施例中，对于类型1，2和3的请求，由“信号”发送类型给定的可以被返回的最大长度分别是268，272或48。然而，I/O总线单元中的工作请求的实际长度是以字节计的，且相关的数据可能相当少。I/O总线单元工作请求指定相关数据的实际长度。

因为“信号”信息可能由于系统排队限制或错误条件而丢失。因而推荐一个管理这些信息的最小的系统队列尺寸。为了在大多数情况下不丢失“信号”信息，对第1类请求至少有17个信号，对第2类请求是24个，对第3类请求是16个。这些数目完全依赖于总线单

元资源支持的特定类型，并且在这里是作为一个较佳实施例对于直接存取存储控制器总线单元给出的。

给出的这些数目是含有 I/O 总线单元工作请求的 I/O 内部缓冲器中的项目的数目。若没有从这些带有主处理器工作请求的缓冲器中清除这些项目，并且这些缓冲器是满的，则将发送一个错误数据类型。“信号”的“偏置”值用作一种跟踪方法以确定“信号”是否丢失了。含有一个非期望的偏置值的主处理器工作请求表明信号信息的丢失。因而，将由 I/O 总线单元来重新发送信号信息。对主处理器的请求的响应包含一个错误码，指明主总线单元应该拒绝该请求。同样的跟踪方法还提供一种相互关系去处理已经发送了要求的记录的定时条件。

主处理器工作请求包含下列信息，

字节	说明
0-1	指令长度
2	指令限定词
3	指令码 = X'23'
4	修改量
	位 0-1 —— 存取方式 = '00'
	位 2-7 —— 保留
	位 2-7 —— 保留
	保留

地址字段是从指定目标的偏置。

字节	说明
8-15	地址, 偏置——带有前导零的 右边对齐
16-23	目标 位16, 信号类型 位17-23, 保留
24-27	活动识别符=总线单元资源识别 符
28-31	GETMAX=信号长度

当从I/O总线单元返回对主处理器的请求的响应时, 它含有说明与该请求相关的数据的下述信息,

字节	说明
0-1	指令长度
2	指令限定词
3	指令码(与信号类型相关)
4	修改量
5-(n-1)	指令正文(n=指令长度)
n-L	数据(L=描述符中数据的 长度)

用DMA请求的反向流程

反向数据流程的控制的另一个方法是使用一对在图28和图29中给出的总线单元信息。服务总线单元为了请求一个DMA进入它的存储, 向请求器发送一个总线单元信息“DMA请求”(DMA

Req) (图28)。从请求总线单元来的总线单元信息“DMA完成”(DMA Comp) (图29)指明DMA操作已完成。该请求总线单元,具有主DMA的总线单元,为服务总线单元完成了一项服务。在请求总线单元中的总线管理不知道哪个OID引起了这个特定的服务请求。

一个例子是服务总线单元中的总线管理收到了一个“运行开始”总线单元信息。然后它发送一个指定请求器存储中的RR0B地址以及它在服务器存储中的位置的“DMA请求”总线单元信息。该请求总线单元将完成此服务,且由一个“DMA完成”总线单元信息通知其它总线单元该操作已完成。接受该DMA请求的总线管理完成所要求的服 务,而不知道该服务的实际请求者是谁。它正在为其它“总线传输机构”服务。

该“DMA请求”有下列字段,

RSVD被保留且必须是零。

“长度”是将以进行“DMA请求”操作而传送的数据的长度。

“DMA识别符”(DMA ID)是用于此“DMA请求”的识别符,并且必须在“DMA完成”总线单元信息中被返回,以确认该特定的DMA请求。在这个DMA请求和DMA完成总线单元信息之外,此识别符没有意义。

“类型识别符(OX)”用来定义总线单元信息的类型以及DMA的方向。一个“DMA请求”有两个可能的16进制值,

! 03 ! —— 从请求处理器到服务处理器

! 04 ! —— 从服务处理器到请求处理器

在请求处理器中的“数据地址”(DATA Address)字段是请求器存储中数据传送的起始地址。类型字段指定数据传送的方向。这可以从“运行开始”总线单元信息得到的R R O B的地址,或者是从该R R O B的内容中得到的数据字段的地址。

在服务处理器中的“数据地址”字段是在该服务器存储中数据传送的起始地址。类型字段指定数据传送方向。

当该请求器完成了所要求的DMA操作时,它通知服务器该操作已完成。在图29中由发送一个“DMA完成”总线单元信息来做到这一点。若在服务处理器执行该要求的DMA操作时发生了一个总线错误,则返回一在“总线错误条件”(BUS ERROR CONDITION)总线单元信息,而不是一个“DMA完成”总线单元信息。

“DMA完成”总线单元信息的各字段如下,

一个“保留”字段必须是零。

“DMA识别符”是“DMA请求”中提供的识别符,它用于确认这一特定DMA请求的请求者。

“类型”字段(05)用来定义总线单元信息的类型。

另两个“保留”字段必须都是零。

由“运行开始”总线单元信息开始的序列将由“运行结束”总线单元信息完成。

图30是一个“发送请求”(SEND REQ)操作的简化的信息流程的例子,该操作起始于一个没有从属DMA能力的总线单元。

1. 服务进程,在这例子中是主总线单元中的一个进程,到达了

在它的处理中它想要等待的那一点处。它做一个“IPCI接收队列”(IPCI RCV Q)(这个例子假定服务器访问接收队列的工作请求)。由于在它的输入队列上没有请求的记录,所以目标进程进入等待状态。

2. 请求器,在这例子中是IOP中的一个进程,想要向一个PU中的服务进程发送数据。该请求进程在IPCI处做一个“IPCI发送请求”(IPCI SEND REQ)。
IPCF总线管理把信息从该进程变换到一个RRCB中。

3. 作为请求器发出“发送请求”的一个结果,IOP中的“总线传输机构”启动总线上的操作。IOP的总线管理向主处理器发送一个“运行开始”总线单元信息。

4. 主处理器中的总线管理发送一个“DMA请求”总线单元信息,它要求IOP中的总线管理向该总线单元信息指定的位置做一次DMA。

5. IOP总线管理做要求的DMA操作,它把RRCB从IOP的存储中移到主处理器的存储中。

6. IOP中的总线管理向主处理器发送一个“DMA完成”总线单元信息,指明要求的DMA操作已经完成。

7. 主处理器中的总线管理发送一个“DMA请求”总线单元信息,它要求IOP中的总线管理做一次从总线管理信息指定的位置开始的DMA操作。这是由主处理器总线管理所拥有的存储。

8. IOP总线管理做要求的DMA操作,把该请求从IOP的存储中移到主处理器的存储中。

9. IOP中的总线管理发送一个“DMA完成”总线单元信

息，指明要求的DMA操作已经完成。若该请求少于4字节，例如RROB中的立即数据，则可不要求上述3个步骤。

10. 通过IPOP向适当的服务进程发送一个“标记”(NOTE)。它符合待完成的接收队列的要求。

11. 现在服务进程必须发送一个“IPCI RCV DTA”来指定把数据放在主处理器存储中的什么地方。

12. 主处理器中的总线管理发送一个“DMA请求”总线单元信息，它要求IOP中的总线管理在该总线单元信息所指定的位置处做一次DMA。该地址由在IPCI处发出的“RCV DTA”来指定。

13. IOP中的总线管理做所要求的DMA操作，把用户数据从IOP的存储中移到主处理器的存储中。

14. IOP中的总线管理发送一个“DMA完成”总线单元信息，指明该请求的DMA操作已经完成。

15. 服务器的“RCV DTA”是符合上述诸操作的要求的。

为传送所有要求的数据，按需要的次数重复步骤12到15。

16. 请求进程，在这例子中是IOP中的一个进程，到达了在它的处理中它想要等待的那一点处。它发送一个“接收队列”。因为在它的输入队列上没有请求，所以请求进程进入等待状态。

17. 现在服务进程必须发送一个带有状态信息的“发送RSP”(SEND RSP)。

18. 主处理器中的总线管理发送一个“运行结束”总线单元信息，指明要求的操作已经完成。

19. 向请求器的队列发送一个“标记”。它满足请求器的“接收队列”(RCV Q)的要求。

通过使用反向流程的方法，保持进程之间通讯的、独立于这些进程并保持这些进程的透明性的目标被保持了。事实上，还使用总线管理把IPCF层次从通讯细节中孤立了出来。使用连接群允许提高流程控制的水平，以保证把资源分配给各进程来提供有保证的最低服务级别。在讨论了一些不同的较佳实施例的同时，很显然本领域的熟练技术人员可以在下列权利要求范围内做出许多变更。

说明书附图

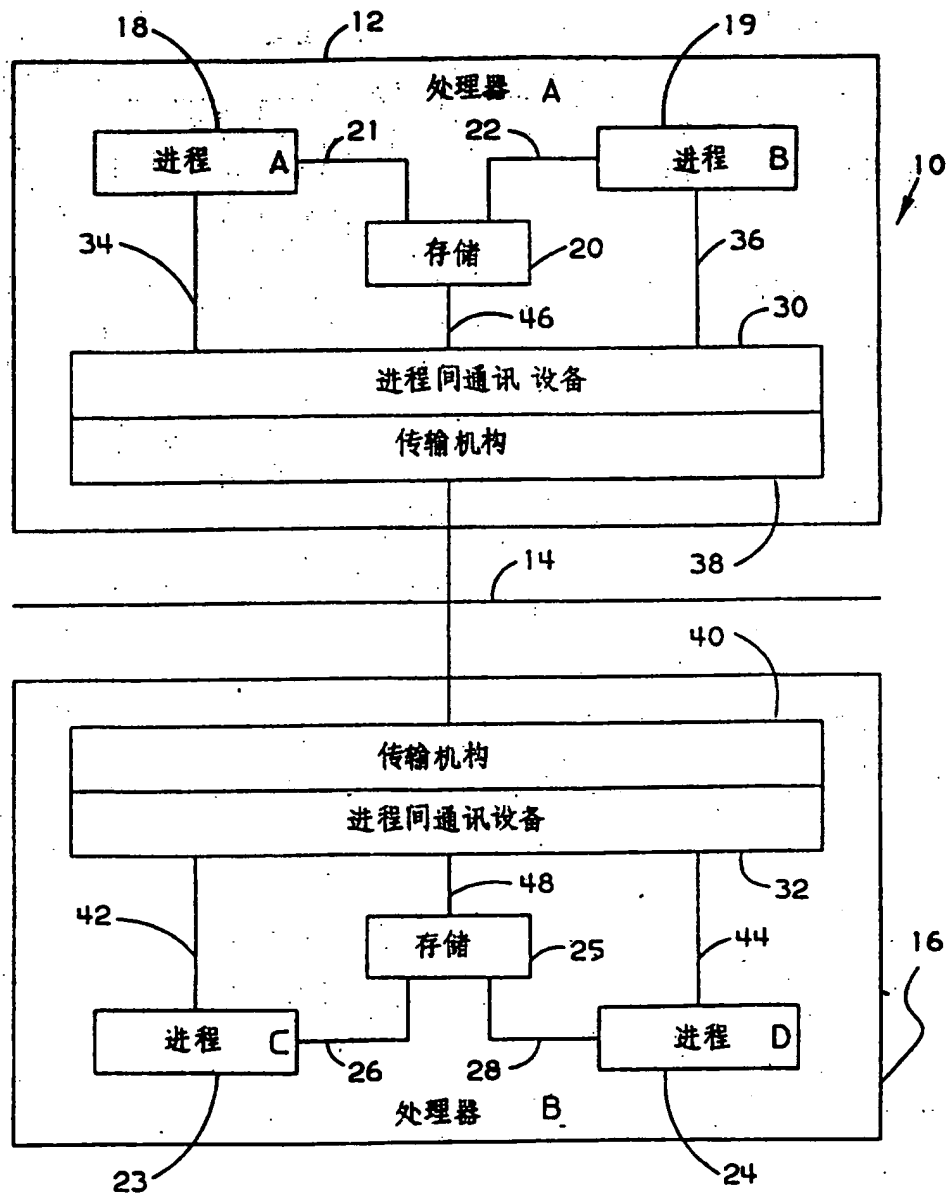


图1

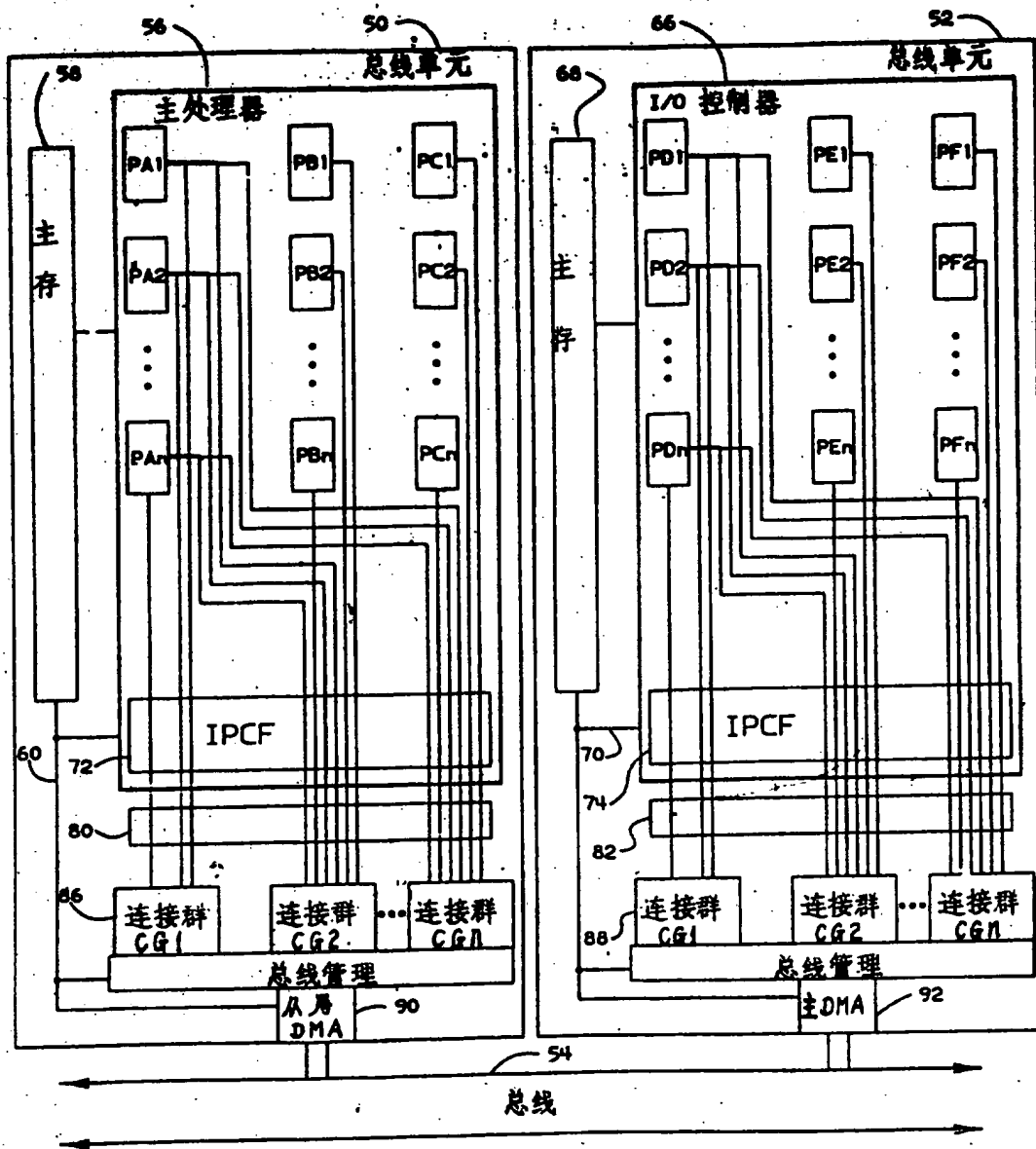


图.2

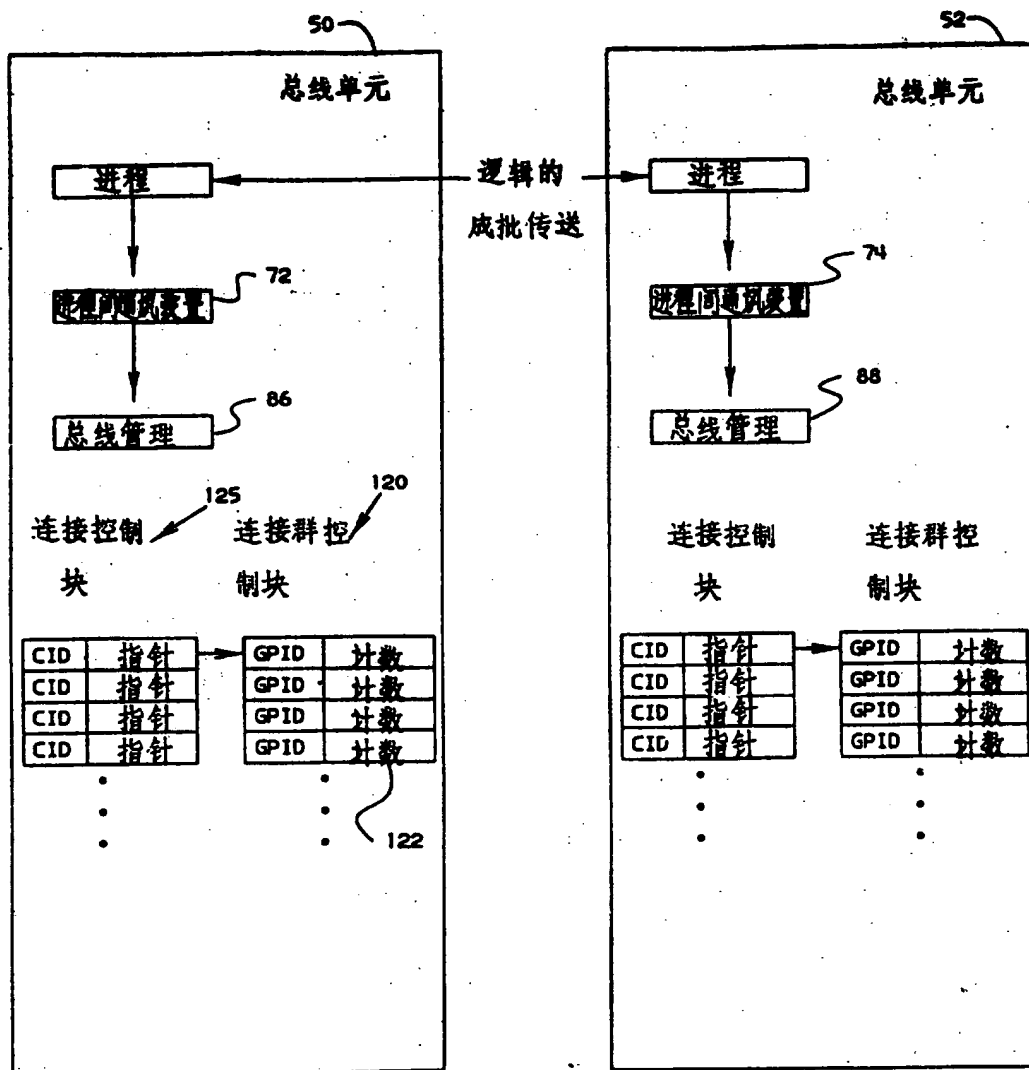


图.3

(请求处理器)

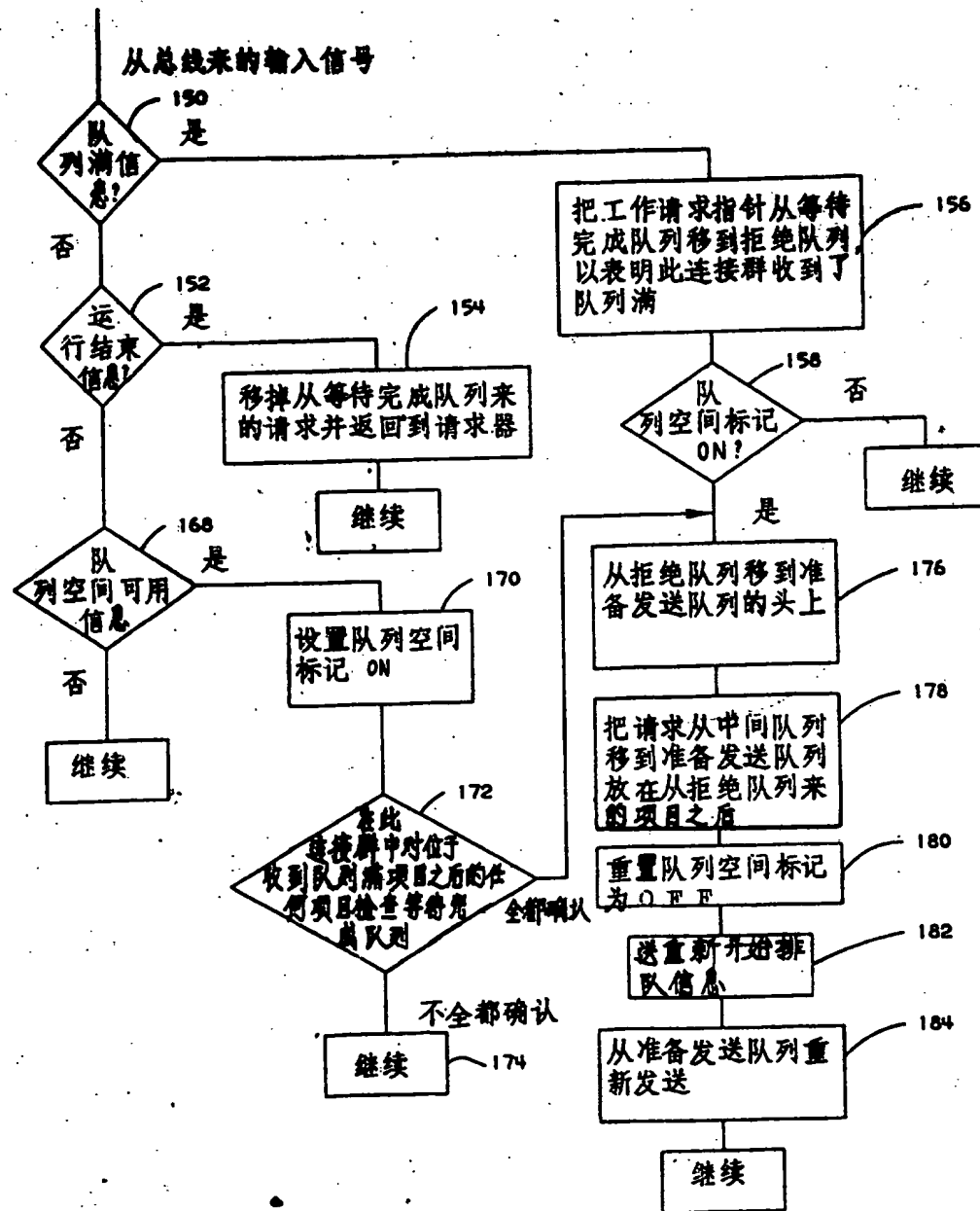


图.4

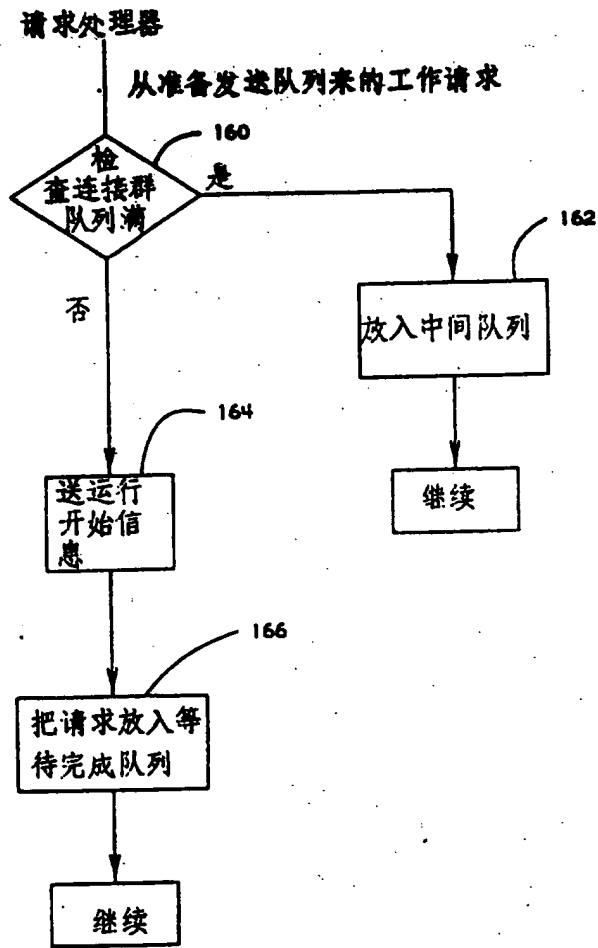


图.5

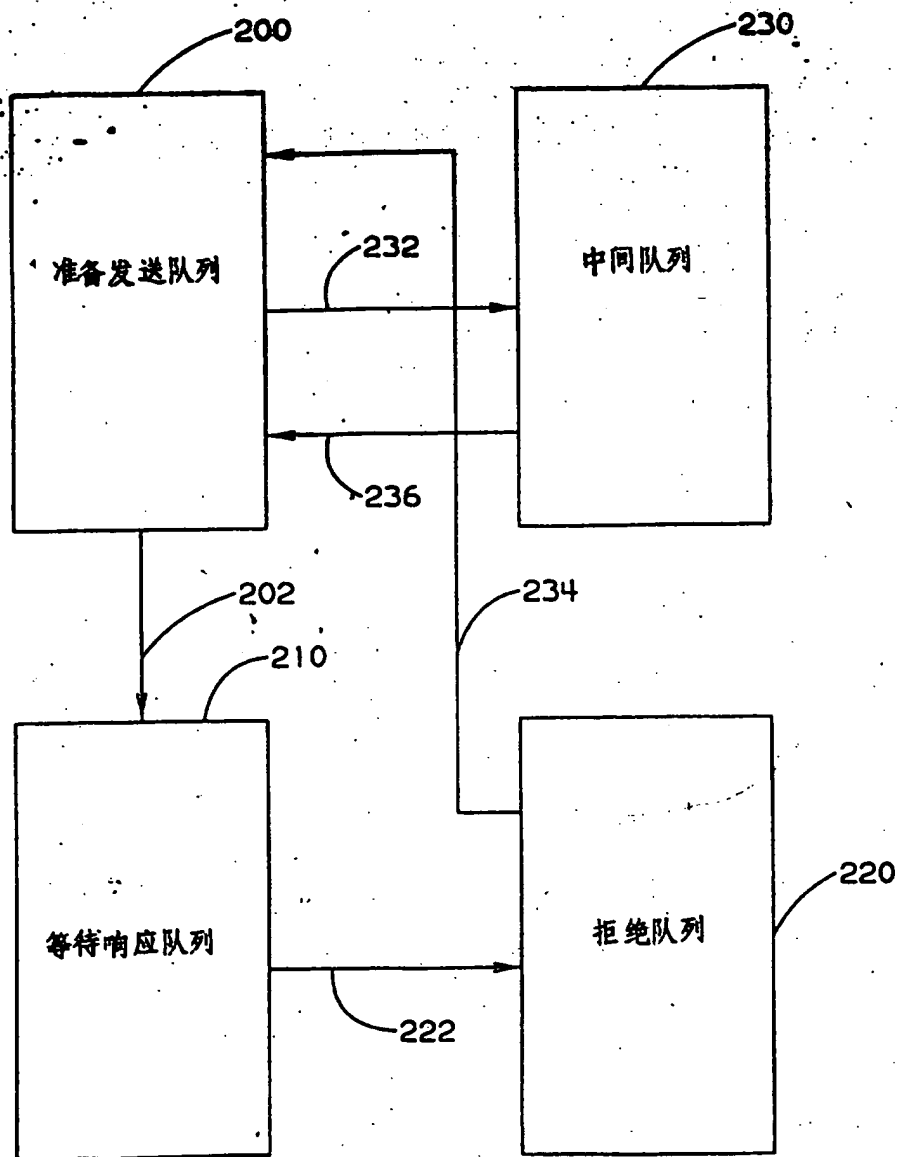


图.6

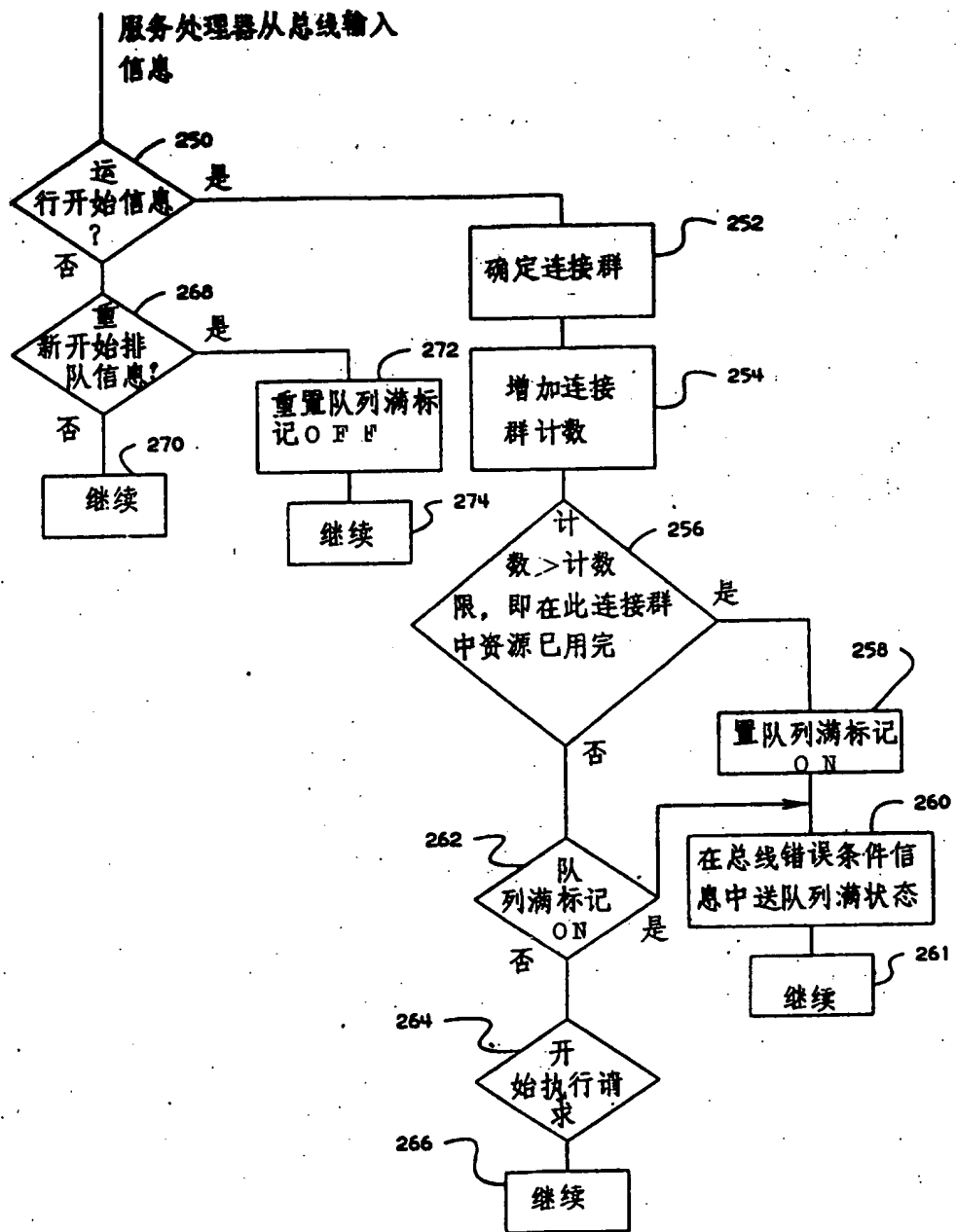


图.7

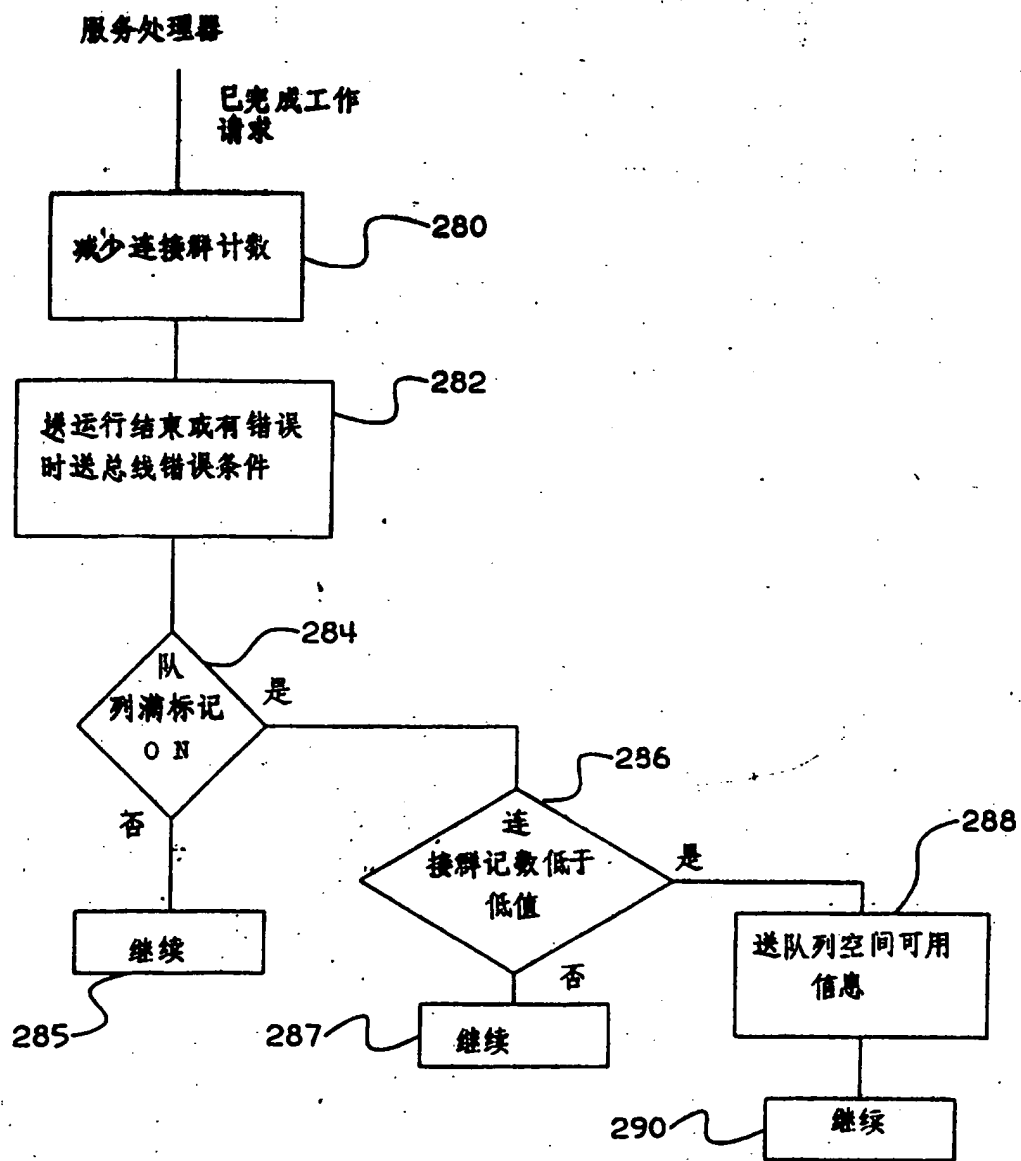


图.8

总线错误条件

保留		总线单元	82
控制字段			
CFID	ACTN	错误状态	连接群识别符

字 1
字 2
字 3

图.9

队列空间可用

保留	保留	OF
保留		群
保留		

字 1
字 2
字 3

图.10

重新开始排队

保留	保留	OD
保留		群
保留		

字1

字2

字3

图.11

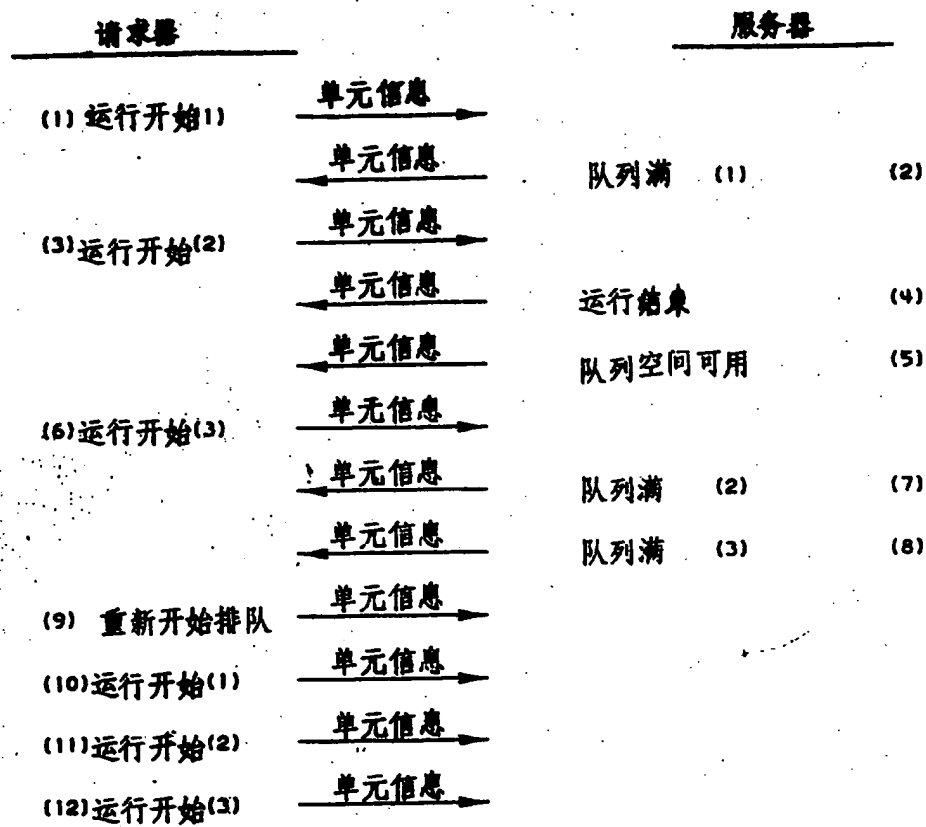


图.12

运行开始

保留	长度	类型
R R C B的地址		
服务器 CID		

字 1

字 2

字 3

图.13

请求/响应控制块

RRCB的长度	保留	类型
服务器连接识别符		
请求的优先级	保留	标记
请求器RID		
保留		
扩展状态指针		
数据标记	数据长度	
数据地址/数据		
数据标记	数据长度	
数据地址/数据		
数据标记	数据长度	
数据地址/数据		
数据标记	数据长度	
数据地址/数据		

字 1

字 2

字 3

字 4

字 5

字 6

字 7

字 8

图 14

附加的请求/响应控制块

数据标记	数据长度
数据地址/数据	
数据标记	数据长度
数据地址/数据	
数据标记	数据长度
数据地址/数据	
数据标记	数据长度
数据地址/数据	

字 1

字 2

图.15

运行结束

保留	标记	02
RID		
完成状态-用户		

字1

字2

字3

图.16

存储请求

保留	保留	06
保留	存储尺寸	
保留	缓冲器长度	

字 1

字 2

字 3

图.17

存储表可用

保留	标记	07
存储表控制块的地址		
保留	长度	

字 1

字 2

字 3

图.18

存储表控制块

保留	总线号	总线单元
缓冲器号	缓冲器长度	
缓冲器地址		
.		
.		
.		

字 1

字 2

字 3

图.19

存储表完成

保留	标记	08
存储表控制块的地址		
保留	(缓冲器的长度)	

字 1

字 2

字 3

图.20

返回存储表

保留	标记	09
请求的存储表的地址		
保留	缓冲器长度	

字1

字2

字3

图.21

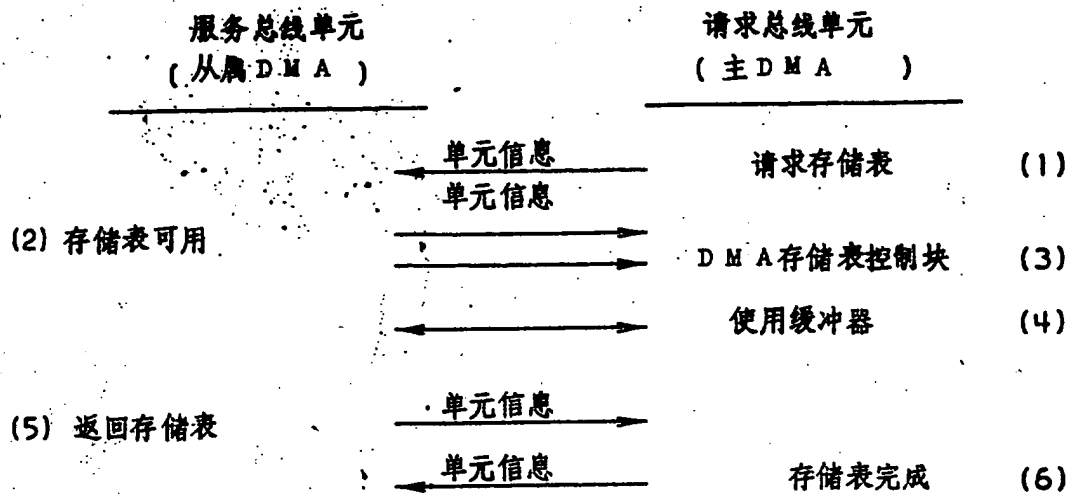


图 22

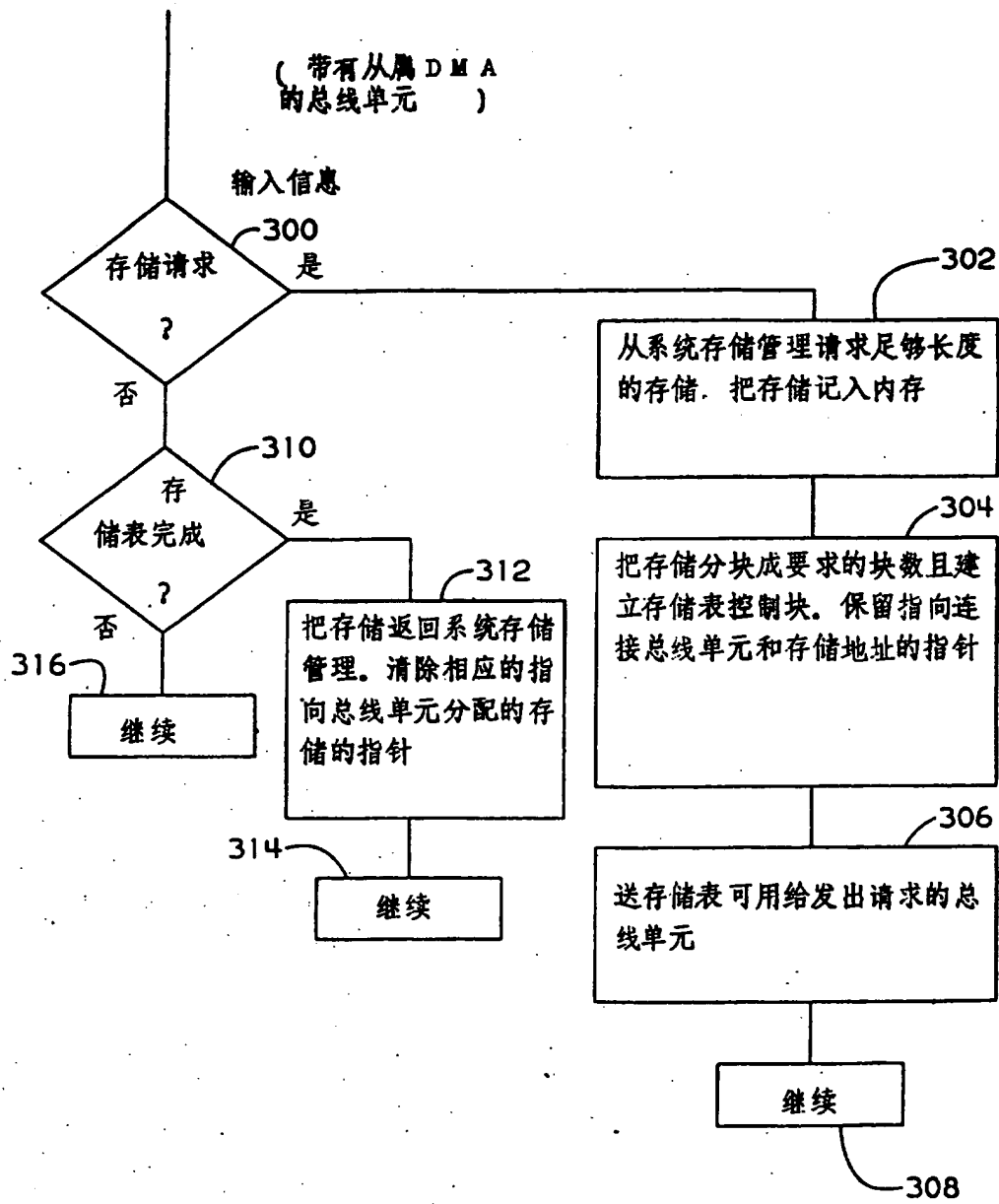


图.23

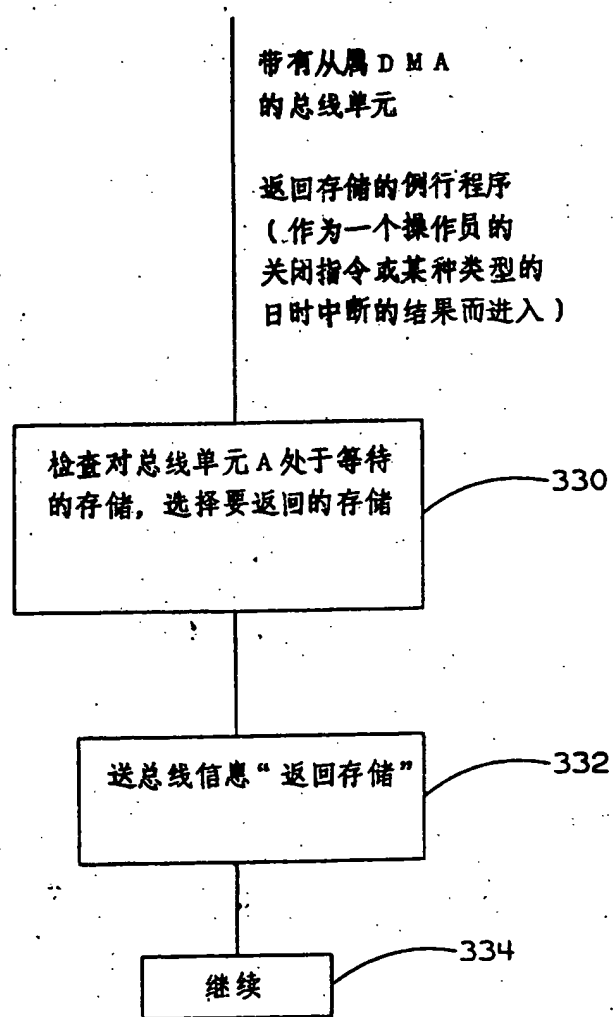


图 24

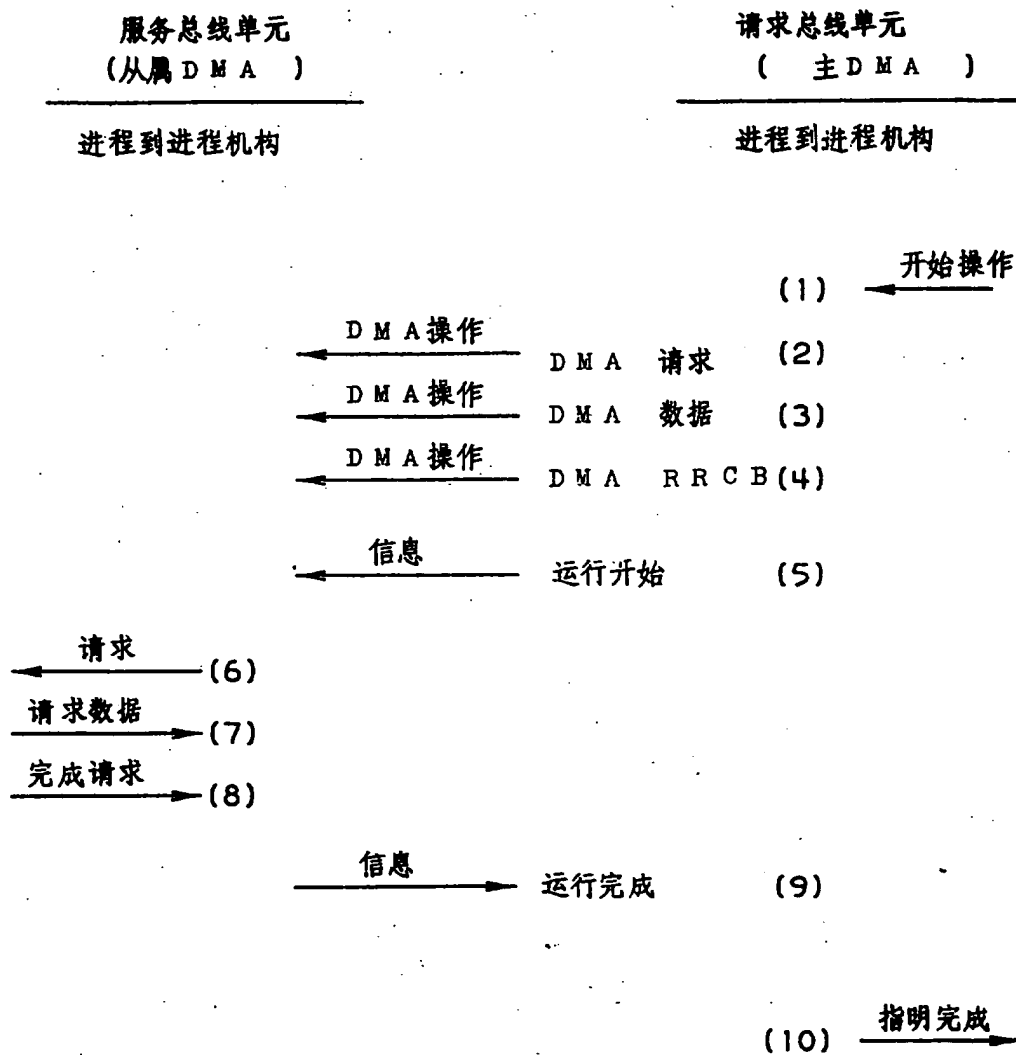


图.25

信号

保留	保留	2X
用户数据		
目标CID		

字1

字2

字3

图.26

另一版本信号

字 1	保留	00	类型
字 2	长度	偏置	
字 3	R A S 连接识别符		

图.27

D M A 请 求

RSVD	长度	识别符	0X
请求处理器中的数据地址			
服务处理器中的数据地址			

字 1

字 2

字 3

图. 28

D M A 完 成

保留	识别符	05
保留		
保留		

字 1

字 2

字 3

图. 29

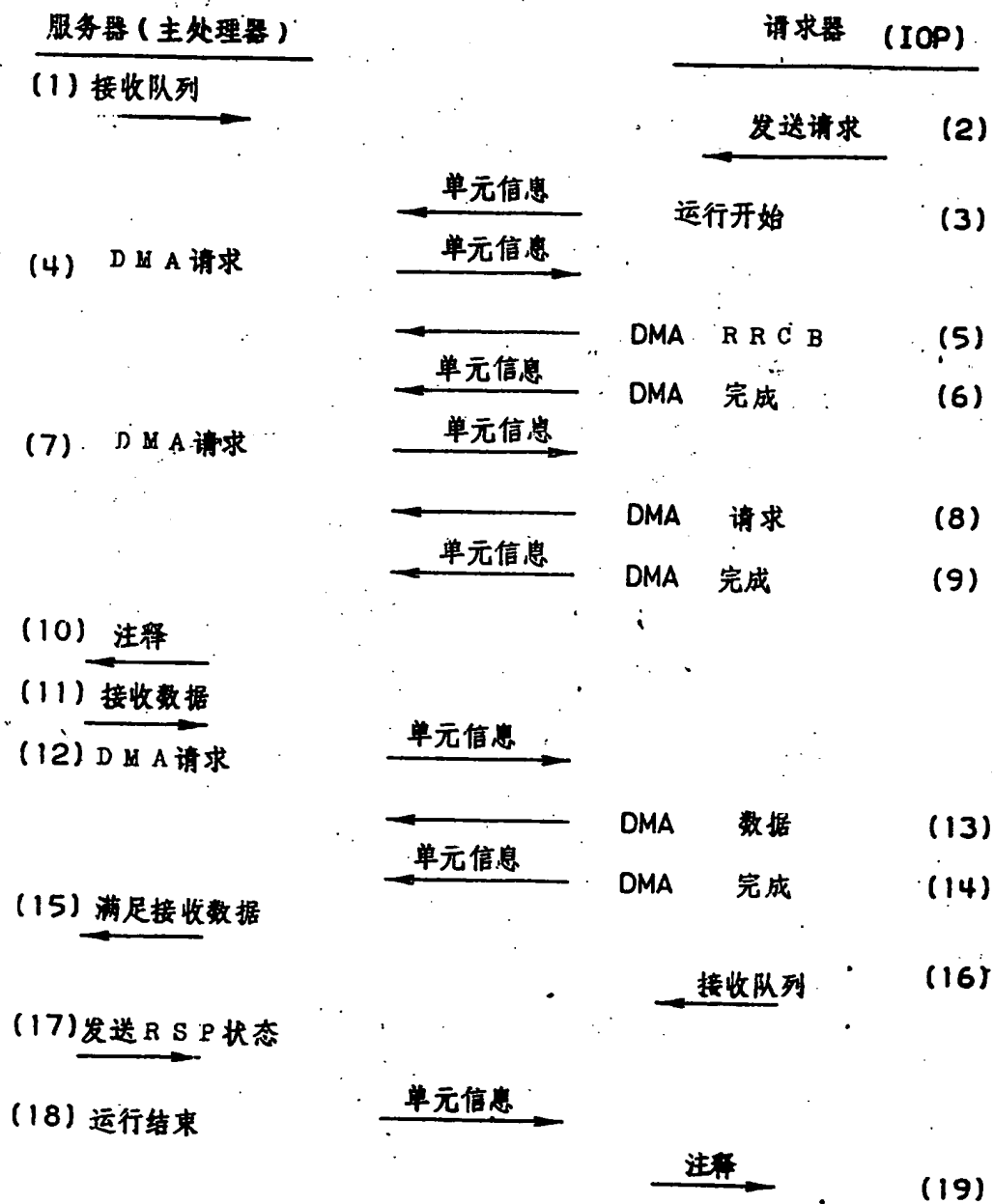


图.30